ELSEVIER

Theoretical Computer Science 211 (1999) 275–309

Theoretical Computer Science

Compositionality properties of *SLD*-derivations

Marco Comini [a,*], Maria Chiara Meo [b]

[a] *Dipartimento di Informatica, Università di Pisa, Corso Italia 40, 56125 Pisa, Italy*
[b] *Dipartimento di Matematica Pura ed Applicata, Università di L'Aquila, via Vetoio, località Coppito, 67010 L'Aquila, Italy*

Received June 1996
Communicated by G. Levi

Abstract

The paper introduces a semantics for definite logic programs expressed in terms of *SLD*-derivations and studies various properties of *SLD*-derivations by using the above semantics.

The semantics of a program is a goal-independent denotation, which can equivalently be specified by a denotational semantics and a transition system. The denotation is proved to be correct, minimal, *AND*-compositional and *OR*-compositional. The denotational semantics and the transition system are defined in terms of a set of primitive semantic operators, whose properties are directly related to the properties of the denotation. The *SLD*-derivations semantics has been designed to act as collecting semantics for a framework of abstract semantics (Comini et al., 1995, 1996).

Keywords: Logic programming; *SLD*-derivations; Semantics; Compositionality

1. Introduction

Lack of compositionality of conventional logic programming semantics has been a serious limitation, since by their very nature PROLOG program fragments are written to be used in an extensible, modular fashion. It has often been noted, in particular, that traditional bottom-up or top-down semantics fail to be sufficiently operational, identify too many computationally distinct programs, and are blind to many interesting observables.

The paper introduces a semantics for definite logic programs expressed in term of *SLD*-derivations and studies various properties of *SLD*-derivations by using the above semantics. The semantics is defined according to the approach in [2], which was already used for some abstractions of *SLD*-derivations, such as computed answers [9], call patterns and partial answers [10] and resultants [11]. The basic idea underlying

* Corresponding author. E-mail: comini@di.unipi.it.

0304-3975/99/$ – see front matter
PII: S0304-3975(97)00174-6

the approach is the *goal-independent program denotation*, which can equivalently be specified by top-down and bottom-up constructions. The top-down definition is the set of *SLD*-derivations for pure atomic goals, while the bottom-up definition is the least fixpoint of a suitable T_P operator. The denotation is proved to be correct and fully abstract w.r.t. the observational equivalence induced on programs by *SLD*-derivations. Moreover, it is proved to enjoy two important compositionality properties, i.e., *AND-compositionality* and *OR-compositionality*. *AND*-compositionality means that the *SLD*-derivations of any goal can be reconstructed from the goal-independent denotation. *OR*-compositionality means that the denotation of $P_1 \cup P_2$ can be reconstructed from the denotations of P_1 and P_2.

The above results simply extend to *SLD*-derivations similar results obtained for other (more abstract) observables. The main novelty of this paper is the semantics definition methodology and the structure of the resulting semantics. We start by defining a *denotational semantics* on domains consisting of sets of *SLD*-derivations. It is a rather standard denotational definition with two peculiarities. First, it deals with low-level operational details, while the usual denotational semantics operates on the domain of computed answers, and are therefore much more abstract. Moreover, the typical compositional style of denotational semantics allows us to identify a small set of *primitive semantic operators*, which are the semantic counterpart of the language syntactic operators. The same primitive semantic operators are then used to define the operational semantics, by means of a *transition system*.

The proof of all the main theorems, such as

- equivalence between denotational and operational semantics,
- equivalence between bottom-up and top-down (goal independent) denotations,
- correctness and minimality of the denotation,
- *AND*-compositionality and *OR*-compositionality of the denotation,

heavily rely on some lemmata, which express properties of the primitive semantic operators. This is even more important, because the *SLD*-derivations semantics has been conceived as the collecting semantics for a hierarchy of semantics [3], systematically derived by using abstract interpretation theory [6]. Since abstraction is essentially abstraction of the primitive semantic operators, the abstract semantics will inherit all those properties of the collecting semantics for which the suitable lemmata on the semantic operators hold. This provides the basis for the definition of a taxonomy of abstractions [3, 4]. It is worth noting that the *SLD*-derivations semantics is the most natural choice for a collecting semantics. It is essentially a traces semantics and it contains all the relevant information of *SLD*-trees. A more abstract semantics, such as the resultant semantics, would not allow to derive properties such as proof trees (used in the Heyting's semantics in [15, 14]) or derivation lengths.

The paper is organized as follows. Section 2 contains background definitions and terminology. Section 3 defines the semantic domain. Section 4 introduces the denotational semantics and the primitive semantic operators. Section 5 defines the transition system. Section 6 defines the goal-independent denotations. Finally, Section 7 contains the main equivalence and compositionality theorems.

2. Preliminaries

In the following sections, we assume familiarity with the standard notions of logic programming as introduced in [1, 17].

Throughout the paper we assume programs and goals being defined on a first-order language given by a signature Σ consisting of a finite set F of *function symbols*, a finite set Π of *predicate symbols* and a denumerable set V of *variable symbols*. T denotes the set of terms built on F and V.

A substitution is a mapping $\vartheta : V \to T$ such that the set $dom(\vartheta) := \{x | \vartheta(x) \neq x\}$ (*domain* of ϑ) is finite. ε is the empty substitution. $range(\vartheta)$ denotes the range of ϑ, i.e., the set $\{y | x \neq \vartheta(x), y \in var(\vartheta(x))\}$. If ϑ is a substitution and E is a syntactic expression, $\vartheta|_E$ is the restriction of ϑ to the variables $var(E)$ of E. The *composition* $\vartheta\sigma$ of the substitutions ϑ and σ is defined as the functional composition. A substitution ϑ is called *idempotent* if $\vartheta\vartheta = \vartheta$ or, equivalently, if $dom(\vartheta) \cap range(\vartheta) = \emptyset$. A *renaming* is a (non-idempotent) substitution ρ for which there exists the inverse ρ^{-1} such that $\rho\rho^{-1} = \rho^{-1}\rho = \varepsilon$.

The preordering $\leqslant$ (more general than) on substitutions is such that $\vartheta \leqslant \sigma$ if and only if there exists ϑ' such that $\vartheta\vartheta' = \sigma$. The result of the application of a substitution ϑ to a term t is an *instance* of t and is denoted by $t\vartheta$. We define $t \leqslant t'$ (t is more general than t') if and only if there exists ϑ such that $t\vartheta = t'$. The relation $\leqslant$ is a preorder (called subsumption) and by $\equiv$ we denote the associated equivalence relation (*variance*). A substitution ϑ is a *unifier* of terms t and t' if $t\vartheta = t'\vartheta$ (where $=$ denotes syntactic equality). If two terms are unifiable then they have an idempotent most general unifier which is unique up to renaming. Therefore, $mgu(t_1, t_2)$ denotes such an idempotent most general unifier of t_1 and t_2. All the above definitions can be extended to other syntactic expressions in the obvious way.

We restrict our attention to idempotent substitutions, unless explicitly stated otherwise. The set of all idempotent substitutions is denoted by *Subst*.

An atom is an object of the form $p(t_1, \ldots, t_n)$ where $p \in \Pi$, $t_1, \ldots, t_n \in T$. A *goal* is a sequence of atoms $A_1, \ldots, A_m$. The empty goal is denoted by $\square$. The set of all atoms is denoted by *Atoms* and the set of all goals is denoted by *Goals*. We denote by $\boldsymbol{G}$ and $\boldsymbol{B}$ possibly empty sequences of atoms, by $\boldsymbol{t}, \boldsymbol{x}$ tuples of, respectively, terms and *distinct* variables. Moreover, we denote by $\boldsymbol{t}$ both the tuple and the set of corresponding syntactic objects. $\boldsymbol{B}, \boldsymbol{B}'$ denotes the concatenation of $\boldsymbol{B}$ and $\boldsymbol{B}'$. An atomic goal is called *pure* if it is in the form $p(\boldsymbol{x})$.

A (definite) *clause* is a formula of the form $H \leftarrow A_1, \ldots, A_n$ with $n \geqslant 0$, where H (the *head*) and $A_1, \ldots, A_n$ (the *body*) are atoms. "$\leftarrow$" and "," denote logical implication and conjunction, respectively, and all variables are universally quantified. If the body is empty the clause is called a *unit clause*. A *program* is a set of (definite) clauses. Given a goal $\boldsymbol{G}$ and a program P, the formula $\boldsymbol{G}$ *in* P (or $P \cup \{\boldsymbol{G}\}$) is a *query*.

Definite clauses have a natural computational reading based on the resolution procedure. The specific resolution strategy called *SLD* can be described as follows. Let $\boldsymbol{G} := A_1, \ldots, A_k$ be a goal and $c := H \leftarrow \boldsymbol{B}$ be a (definite) clause. $\boldsymbol{G}'$ is *derived* from

$\boldsymbol{G}$ and c by using ϑ if and only if there exists an atom A_m, $1 \leqslant m \leqslant k$ such that $\vartheta = mgu(A_m, H)$ and $\boldsymbol{G}' = (A_1, \ldots, A_{m-1}, \boldsymbol{B}, A_{m+1}, \ldots, A_k)\vartheta$. An *SLD-derivation* (or simply a derivation) of the query $\boldsymbol{G}$ *in* P consists of a (possibly infinite) sequence of goals $\boldsymbol{G}_0, \boldsymbol{G}_1, \boldsymbol{G}_2, \ldots$ called resolvents, together with a sequence $c_1, c_2, \ldots$ of variants of clauses in P which are *renamed apart*[1] and a sequence $\vartheta_1, \vartheta_2, \ldots$ of idempotent *mgus* such that $\boldsymbol{G}_0 = \boldsymbol{G}$ and, for $i \geqslant 1$, each $\boldsymbol{G}_i$ is derived from $\boldsymbol{G}_{i-1}$ and c_i by using ϑ_i. An *SLD-refutation* of $\boldsymbol{G}$ *in* P is a finite *SLD*-derivation of $\boldsymbol{G}$ *in* P which has the empty goal $\square$ as the last goal in the derivation. An *SLD-tree* of $\boldsymbol{G}$ *in* P is the prefix tree of all *SLD*-derivations of $\boldsymbol{G}$ *in* P.

A *selection rule* R is a function which, when applied to a "history" containing the goal, all the clauses and the *mgus* used in the derivation $\boldsymbol{G}_0, \boldsymbol{G}_1, \ldots, \boldsymbol{G}_i$, returns an atom in $\boldsymbol{G}_i$. Such an atom is the selected atom in $\boldsymbol{G}_i$. In the following, for the sake of simplicity, we consider the PROLOG leftmost selection rule. All our results can be generalized to skeleton rules [11].

In the following $\boldsymbol{G} \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_n]{\vartheta_n} \boldsymbol{G}_n$ $(n \geqslant 0)$, denotes a (partial and finite) *SLD*-derivation of goal $\boldsymbol{G}$ via the leftmost selection rule. The derivation uses the renamed apart clauses $c_1, \ldots, c_n$ and $\vartheta := (\vartheta_1 \cdots \vartheta_n)|_{\boldsymbol{G}}$ is the computed answer substitution of $\boldsymbol{G}$. We also denote by $\boldsymbol{G} \xrightarrow[P]{\vartheta}{}^{*} \boldsymbol{B}$ a finite *SLD*-derivation of $\boldsymbol{G}$ *in* P via the leftmost selection rule, where ϑ is the computed answer substitution and $\boldsymbol{B}$ is the last resolvent.

Given a derivation d, *first*(d) and *last*(d) (if d is finite) are, respectively, the first and the last goal of d. *length*(d) denotes the length of the derivation and *clauses*(d) denotes the sequence of clauses of d. Moreover, *prefix*(d) is the set of all derivations which are prefixes of d. By an abuse of notation, we denote a zero-length derivation of $\boldsymbol{G}$ by $\boldsymbol{G}$ itself.

In the paper we use standard results on the ordinal powers $\uparrow n$ of continuous functions on the complete lattices. Namely, given any monotonic operator T on $(C, \leqslant)$, $T\uparrow\omega := \bigsqcup_{n<\omega} T\uparrow n$, $T\uparrow n+1 := T(T\uparrow n)$, for $n < \omega$, and $T\uparrow 0 := \bot_C$, where $\bot_C$ is the least element of C. Moreover, if T is continuous its least fixpoint is $T\uparrow\omega$.

We use lambda notation to denote partial functions by allowing expressions in lambda terms that are not always defined. Hence, a lambda expression $\lambda x.E$ denotes a partial function which on input x assumes the value $E[x]$ if the expression $E[x]$ is defined, otherwise it is undefined. $g := f\left[{}^{v}/_{x}\right]$ denotes the function g such that $g(x) = v$ and $\forall y \neq x. g(y) = f(y)$. Furthermore, $\bot$ denotes the undefined element. For each set S we define $\bot \subseteq S$ and $\bot \cup S = S$. Note that $\emptyset \not\subseteq \bot$.

3. Semantic domains

We assume the reader to be familiar with the notions of *SLD*-resolution and *SLD*-tree [17, 1]. We represent here, for notational convenience, *SLD*-trees as sets of derivations.

[1] i.e., such that c_i does not share any variable with $\boldsymbol{G}_0$, $c_1, \ldots, c_{i-1}$.

(1) A set of derivations S is *well-formed* if and only if for any $\mathsf{d} \in S$ we have $\mathit{prefix}(\mathsf{d}) \subseteq S$. Each well-formed set where all the clauses used in derivations are in P is a representation of a family of partial *SLD*-trees of P.
(2) We denote by *WFS* the complete lattice of well-formed sets of derivations, partially ordered by $\subseteq$. The maximal well-formed set of derivations of $\boldsymbol{G}$ *in* P is a representation of the *SLD*-tree of $\boldsymbol{G}$ *in* P.
(3) A *collection* D is a partial function $\mathit{Goals} \longrightarrow \mathit{WFS}$ such that, for every $\boldsymbol{G} \in \mathit{Goals}$, if $D(\boldsymbol{G})$ is defined, then it is a well-formed set of derivations *all starting from* $\boldsymbol{G}$. Formally, $\forall \mathsf{d} \in D(\boldsymbol{G}).\ \mathit{first}(\mathsf{d}) = \boldsymbol{G}$. Hence, a collection is a function which associates to any goal $\boldsymbol{G}$ a (representation of) a partial *SLD*-tree of $\boldsymbol{G}$ *in* P. A *pure* collection is a collection defined only for pure atomic goals.
(4) $\mathbb{C}$ is the domain of all the collections ordered by $\sqsubseteq$, where $D \sqsubseteq D'$ if and only if $\forall \boldsymbol{G}.\ D(\boldsymbol{G}) \subseteq D'(\boldsymbol{G})$. The partial order on $\mathbb{C}$ formalizes the evolution of the computation process. It is easy to prove that $(\mathbb{C}, \sqsubseteq)$ is a complete lattice. $\mathbb{PC}$ is the sub-lattice of all pure collections.

The goal we want to achieve is to develop a denotation modeling partial *SLD*-trees. We follow the approach in [2], by defining a "syntactic" semantic domain (interpretation). Our modeling of partial *SLD*-trees is similar to the basic denotation defined in terms of clauses in [10].

In order for the semantics not to depend upon variable names and on the specific unification algorithm, we define the *equivalence modulo enhanced variance* $\equiv_{\mathbb{C}}$ on collections. Namely, $D \equiv_{\mathbb{C}} D'$ if and only if, for any $\boldsymbol{G}$, there exists a variant $\boldsymbol{G}'$ of $\boldsymbol{G}$ such that, if $D(\boldsymbol{G})$ is defined, then $D'(\boldsymbol{G}')$ is defined and, for any $\mathsf{d} \in D(\boldsymbol{G})$, there exists $\mathsf{d}' \in D'(\boldsymbol{G}')$, such that $\mathit{clauses}(\mathsf{d}) \equiv \mathit{clauses}(\mathsf{d}')$ and vice versa. Hence, derivations with a different choice of the *mgu* and of the new variables introduced by the renaming apart operation are equivalent modulo enhanced variance.

Definition 1. An interpretation I ($\mathbb{C}$-interpretation) is a pure collection modulo enhanced variance. We denote by $\mathbb{I}_{\mathbb{C}}$ the set of interpretations. $(\mathbb{I}_{\mathbb{C}}, \sqsubseteq)$ is a complete lattice with the induced quotient order.

Note that in interpretations the enhanced variance relation allows us to abstract w.r.t. the variables occurring in the initial goals of any collection.

4. Denotational semantics of *SLD*-derivations

We start with some notation. We denote the equivalence class (modulo enhanced variance) of a collection σ by σ itself. Moreover, any interpretation I of $\mathbb{I}_{\mathbb{C}}$ is implicitly considered also as an arbitrary collection obtained by choosing an arbitrary representative of I. All the semantic operators that we use on interpretations are independent of the choice of the representative. Therefore, we can define any operator on $\mathbb{I}_{\mathbb{C}}$ in terms of its counterpart defined on $\mathbb{C}$, independently from the choice of the representative.

All the definitions are independent from the choice of the syntactic object. To simplify the notation, we denote the corresponding operators on $\mathbb{I}_{\mathbb{C}}$ and $\mathbb{C}$ by the same name.

Several denotational semantics have been defined for logic programs (see, for example, [7, 12, 13]). The main differences w.r.t. the above definitions are that we do not consider the PROLOG search rule and that our denotations model *SLD*-derivations rather than just computed answers.

We define the denotational semantics inductively on the following syntax of logic programs (the syntactic structure of atoms is not defined).

$$\begin{aligned} QUERY &::= GOAL \ in \ PROG \\ GOAL &::= \Box \mid ATOM,\ GOAL \\ PROG &::= \emptyset \mid \{CLAUSE\} \cup PROG \\ CLAUSE &::= ATOM \leftarrow GOAL \end{aligned}$$

The semantic functions are

$$\begin{aligned} &\mathscr{Q} : QUERY \longrightarrow \mathbb{C}, \\ &\mathscr{G} : GOAL \longrightarrow (\mathbb{I}_{\mathbb{C}} \to \mathbb{C}), \qquad \mathscr{A} : ATOM \longrightarrow (\mathbb{I}_{\mathbb{C}} \to \mathbb{C}), \\ &\mathscr{P} : PROG \longrightarrow (\mathbb{I}_{\mathbb{C}} \to \mathbb{I}_{\mathbb{C}}), \qquad \mathscr{C} : CLAUSE \longrightarrow (\mathbb{I}_{\mathbb{C}} \to \mathbb{I}_{\mathbb{C}}) \end{aligned}$$

and are defined in terms of the semantic operators $\cdot$, $\times$, $\bowtie$, $\sum$ defined in Section 4.2. The choice of the semantic operators is induced by syntactic operations, due to the compositional nature of definitions in the denotational style. The informal meaning of the operators is the following. The operator $\cdot$ "solves" an atomic goal A into an interpretation I. The operator $\times$ computes the conjunction of two interpretations. The operator $\bowtie$ computes the interpretation obtained by replacement. The operator $\sum$ computes the non-deterministic union of a class of interpretations. Note that when the class is finite we use the infix notation $+$. Finally, the function $\phi_{\Box}$ is the collection of the empty goal and *tree* maps clauses to collections.

$$\mathscr{Q}[\![\boldsymbol{G} \ in \ P]\!] := \mathscr{G}[\![\boldsymbol{G}]\!]_{lfp\,\mathscr{P}[\![\mathscr{P}]\!]} \tag{1}$$

$$\mathscr{G}[\![A, \boldsymbol{G}]\!]_I := \mathscr{A}[\![\mathscr{A}]\!]_I \times \mathscr{G}[\![\boldsymbol{G}]\!]_I \quad \mathscr{G}[\![\Box]\!]_I := \phi_{\Box} \tag{2}$$

$$\mathscr{A}[\![A]\!]_I := A \cdot I \tag{3}$$

$$\mathscr{P}[\![\{c\} \cup P]\!]_I := \mathscr{C}[\![c]\!]_I + \mathscr{P}[\![P]\!]_I \quad \mathscr{P}[\![\emptyset]\!]_I := Id_{\mathbb{I}} \tag{4}$$

$$\mathscr{C}[\![p(\boldsymbol{t}) \leftarrow \boldsymbol{B}]\!]_I := tree(p(\boldsymbol{t}) \leftarrow \boldsymbol{B}) \bowtie \mathscr{G}[\![\boldsymbol{B}]\!]_I, \tag{5}$$

where $lfp\,\mathscr{P}[\![P]\!]$ means $lfp_{\mathbb{I}_{\mathbb{C}}} \lambda I.\mathscr{P}[\![P]\!]_I$.

The last definition (5) evaluates to the (collection mapping the pure version of the head of the clause to the) one-step derivation using the clause $p(\boldsymbol{t}) \leftarrow \boldsymbol{B}$ followed by all (suitably renamed) derivations starting with $\boldsymbol{B}$ obtained by composition from I.

```
c1: ap([], Xs, Xs).
    c2: ap([X|Xs], Ys, [X|Zs]) :- ap(Xs, Ys, Zs).
```

Fig. 1. The append program.

Example 2. Consider the (well-known *append*) program P of Fig. 1 and the goal $\boldsymbol{G} := ap([\mathsf{a}],[l],x), ap(x,[h],z)$. Then the denotation of $\boldsymbol{G}$ *in* P is

$$\begin{aligned}\mathscr{D}[\![\boldsymbol{G} \text{ in } P]\!] &= \mathscr{G}[\![\boldsymbol{G}]\!]_{lfp\mathscr{P}[\![P]\!]}\\ &= \mathscr{A}[\![ap([\mathsf{a}],[l],x)]\!]_{lfp\mathscr{P}[\![P]\!]} \times \mathscr{A}[\![ap(x,[h],z)]\!]_{lfp\mathscr{P}[\![P]\!]} \times \phi_{\square}\\ &= \big(ap([\mathsf{a}],[l],x)\cdot lfp\mathscr{P}[\![P]\!]\big) \times \big(ap(x,[h],z)\cdot lfp\mathscr{P}[\![P]\!]\big)\end{aligned}$$

Then, since

$$\mathscr{P}[\![P]\!]_I(ap(x,y,z)) = \left\{ ap(x,y,z),\quad ap(x,y,z)\xrightarrow[ap([\,],v,v)]{\{x/[\,],y/v,z/v\}}\square,\right.$$

$$\left. ap(x,y,z)\xrightarrow[ap([l|u],t,[l|v])\leftarrow ap(u,t,v)]{\{x/[l|u],y/t,z/[l|v]\}} ap(u,t,v)\right\} \bowtie (ap(u,t,v)\cdot I)$$

$$lfp\mathscr{P}[\![P]\!](ap(x,y,z)) = \left\{ ap(x,y,z),\quad ap(x,y,z)\xrightarrow[ap([\,],v,v)]{\{x/[\,],y/v,z/v\}}\square,\right.$$

$$ap(x,y,z)\xrightarrow[ap([l|u],t,[l|v])\rightarrow ap(u,t,v)]{\{x/[l|u],y/t,z/[l|v]\}} ap(u,t,v),$$

$$\left. ap(x,y,z)\xrightarrow[ap([l|u],t,[l|v])\leftarrow ap(u,t,v)]{\{x/[l|u],y/t,z/[l|v]\}} ap(u,t,v)\xrightarrow[ap([\,],w,w)]{\{u/[\,],t/w,v/w\}}\square,\ldots\right\},$$

the following hold:

$$\big(ap([\mathsf{a}],[l],x)\cdot lfp\mathscr{P}[\![P]\!]\big)(ap([\mathsf{a}],[l],x)) = prefix\Big(ap([\mathsf{a}],[l];x)$$

$$\xrightarrow[ap([r|y],v,[r|w])\leftarrow ap(y,v,w)]{\{x/[\mathsf{a}|w],v/[l],y/[\,],r/\mathsf{a}\}} ap([\,],[l]w)\xrightarrow[ap([\,],t,t)]{\{w/[l],t/[l]\}}\square\Big)$$

$$\big(ap(x,[h],z)\cdot lfp\mathscr{P}[\![P]\!]\big)(ap(x,[h],z)) = prefix\left(ap(x,[h],z)\xrightarrow[ap([\,],y,y)]{\{x/[\,],z/[h],y/[h]\}}\square\right)$$

$$\cup prefix\Big(ap(x,[h],z)\xrightarrow[ap([l|y],t,[l|v])\leftarrow ap(y,t,v)]{\{x/[l|y],t/[h],z/[l|v]\}}$$

$$ap(y,[h],v)\xrightarrow[ap([\,],u,u)]{\{y/[\,],u/[h],v/[h]\}}\square\Big)\cup\cdots$$

$$\mathscr{D}[\![G \text{ in } P]\!](G) = prefix\Big(ap([\mathsf{a}],[l],x),\ ap(x,[h],z) \xrightarrow[ap([r|y],v,[r|w]) \leftarrow ap(y,v,w)]{\{x/[\mathsf{a}|w],\ v/[l],\ y/[\,],\ r/\mathsf{a}\}}$$

$$ap([\,],[l],w),\ ap([\mathsf{a}|w],[h],z) \xrightarrow[ap([\,],t,t)]{\{w/[l],\ t/[l]\}} ap([\mathsf{a},l],[h],z)$$

$$\xrightarrow[ap([o|v],u,[o|s]) \leftarrow ap(v,u,s)]{\{o/\mathsf{a},\ v/[l],\ u/[h],\ z/[\mathsf{a}|s]\}} ap([l],[h],s)$$

$$\xrightarrow[ap([o'|v'],u',[o'|s']) \leftarrow ap(v',u',s')]{\{o'/l,\ v'/[\,],\ u'/[h],\ s/[l|s']\}} ap([\,],[h],s') \xrightarrow[ap([\,],t',t')]{\{t'/[h],\ s'/[h]\}} \square\Big).$$

4.1. Basic operators on derivations

In order to define the semantic operators we first need to define three auxiliary operations on derivations. Such operators will be used later to define the semantics operations on interpretations. The first operation formalizes the concatenation of two derivations, the second computes the instantiation of a derivation, and finally the third defines the *AND*-compositional conjunction of two derivations. These are the formal definitions.

(1) Let d_1, d_2 be derivations such that $last(\mathsf{d}_1) = first(\mathsf{d}_2)$ and $var(\mathsf{d}_1) \cap var(\mathsf{d}_2) = var(first(\mathsf{d}_2))$. Then $\mathsf{d}_1 :: \mathsf{d}_2$ denotes the concatenation of d_1 and d_2.

(2) Let $\mathsf{d} := G'_0 \xrightarrow[c_1]{\vartheta'_1} \cdots \xrightarrow[c_k]{\vartheta'_k} G'_k$ be a derivation and γ be an idempotent substitution such that $var(G'_0\gamma) \cap var(clauses(\mathsf{d})) = \emptyset$. Then $\partial_\gamma(\mathsf{d}) := G_0 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_h} G_h$ where
- $G_0 := G'_0\gamma$ and
- for any $0 < i \leqslant k$, if $G_{i-1} = (A, \bar{G}_i)$ and $c_i = H \leftarrow B$ then (if an *mgu* exists) $\vartheta_i := mgu(A,H)$ and $G_i := (B, \bar{G}_i)\vartheta_i$.

Note that $\partial_\gamma(\mathsf{d})$ is the derivation obtained by applying the substitution γ to $first(\mathsf{d})$ and attempting to build as long a derivation as possible (until failure to find *mgu* impedes it) using the same clauses as in d. Thus, in particular $h \leqslant k$.

(3) Let $\mathsf{d}_1 := G'_0 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} G'_k$, d_2 be derivations such that $G''_0 = first(\mathsf{d}_2)$ and $var(\mathsf{d}_1) \cap var(\mathsf{d}_2) = var(G'_0) \cap var(G''_0)$. Then $\mathsf{d}_1 \wedge \mathsf{d}_2$ is defined as follows: if $G'_k \neq \square$ then $\mathsf{d}_1 \wedge \mathsf{d}_2 := (G'_0, G''_0) \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} (G'_k, G''_0\vartheta_1 \cdots \vartheta_k)$, otherwise is $\Big((G'_0, G''_0) \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} G''_0\vartheta_1 \cdots \vartheta_k\Big) :: \partial_{\vartheta_1 \cdots \vartheta_k}(\mathsf{d}_2)$. Note that $\mathsf{d}_1 \wedge \mathsf{d}_2$ is the derivation obtained by adding (a suitable instantiation of) the goal $first(\mathsf{d}_2)$ to each goal in d_1 and then (if d_1 is a refutation) attempting to build as long a derivation as possible using the same clauses as in d_2.

The constraints on the variables of derivations are used to avoid variable name clashes in the clauses. Moreover, note that for any choice of the *mgu* used in the construction of the derivations, the results are equivalent modulo variance.

The following lemma states that the above operations are well defined.

Lemma 3. *Let* d_1 *and* d_2 *be derivations and* γ *be an idempotent substitution. Then the following properties hold.*
(1) *If* $\mathsf{d}_1 :: \mathsf{d}_2$ *is defined then* $\mathsf{d}_1 :: \mathsf{d}_2$ *is a derivation.*
(2) *If* $\partial_\gamma(\mathsf{d}_1)$ *is defined, then* $\partial_\gamma(\mathsf{d}_1)$ *is a derivation.*
(3) *If* $\mathsf{d}_1 \wedge \mathsf{d}_2$ *is defined then* $\mathsf{d}_1 \wedge \mathsf{d}_2$ *is a derivation.*

Lemma 4. *Let* $\mathsf{d}_1, \mathsf{d}_2$ *and* d_3 *be derivations and* γ *be an idempotent substitution. Then the following holds*:
(1) *If* $\partial_\gamma(\mathsf{d}_1 \wedge \mathsf{d}_2)$ *and* $\partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2)$ *are defined then* $\partial_\gamma(\mathsf{d}_1 \wedge \mathsf{d}_2) = \partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2)$.
(2) *If* $(\mathsf{d}_1 \wedge \mathsf{d}_2) \wedge \mathsf{d}_3$ *and* $\mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3)$ *are defined then* $(\mathsf{d}_1 \wedge \mathsf{d}_2) \wedge \mathsf{d}_3 = \mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3)$.

4.2. Basic operators on collections

Let D, D_1 and D_2 be collections in $\mathbb{C}$, $\boldsymbol{G}$ be a goal and A be an atom.

The *void collection* ϕ is the collection $\lambda \boldsymbol{G}.\bot$, i.e., the undefined function.

The *identity collection* $Id_{\mathbb{C}}$ is the collection of zero-length derivations for each goal, i.e., $\lambda \boldsymbol{G}.\{\boldsymbol{G}\}$, while the *pure* identity collection $Id_{\mathbb{I}}$ is the collection $\lambda p(\boldsymbol{x}).\{p(\boldsymbol{x})\}$. Moreover, $\phi_{\boldsymbol{G}}$ denotes the collection $\phi\left[{}^{\{\boldsymbol{G}\}}/_{\boldsymbol{G}}\right]$.[2]

The *instantiation* of D with A is

$$A \cdot D := \phi\left[{}^{S}/_{A}\right] \text{ where}$$
$$S := \{\partial_\gamma(\mathsf{d}) \mid S' \text{ is a renamed apart (from } A\text{) version of } D(A'), \text{ for some } A' \leqslant A,\ \mathsf{d} \in S' \text{ and there exists } \gamma \text{ such that } A = \mathit{first}(\mathsf{d})\gamma\}.$$

The *product* of D_1 and D_2 is

$$D_1 \times D_2 := \lambda \boldsymbol{G}.\{\mathsf{d}_1 \wedge \mathsf{d}_2 \mid (\boldsymbol{G}_1, \boldsymbol{G}_2) = \boldsymbol{G} \text{ and for } i = 1,2,\ \boldsymbol{G}_i' \equiv \boldsymbol{G}_i, \mathsf{d}_i \text{ is a renamed version of an element in } D_i(\boldsymbol{G}_i'), \text{ such that } \boldsymbol{G}_i = \mathit{first}(\mathsf{d}_i) \text{ and } \mathsf{d}_1 \wedge \mathsf{d}_2 \text{ is defined}\}.$$

The (*compatible*) *extension* of D_1 by D_2 is

$$D_1 \bowtie D_2 := \lambda \boldsymbol{G}.D_1(\boldsymbol{G}) \cup \{\mathsf{d}_1 :: \mathsf{d}_2 \mid \mathsf{d}_1 \in D_1(\boldsymbol{G}), \boldsymbol{G}_2 \equiv \mathit{last}(\mathsf{d}_1) \text{ and } \mathsf{d}_2 \text{ is a renamed version of an element in } D_2(\boldsymbol{G}_2), \text{ such that } \mathsf{d}_1 :: \mathsf{d}_2 \text{ is defined}\}.$$

The $\bowtie$ operator is extensive on the first argument, i.e., $D_1 \sqsubseteq D_1 \bowtie D_2$.
The *sum* of a class $\{D_j\}_{j \in J}$ is

$$\sum\{D_j\}_{j \in J} := \lambda \boldsymbol{G}.\bigcup_{j \in J} D_j(\boldsymbol{G}).$$

[2] Note that the void collection, viewed as a set of ordered pairs, is just the empty set and $\phi_{\boldsymbol{G}}$ is just the singleton $\{\langle \boldsymbol{G}, \{\boldsymbol{G}\}\rangle\}$.

Note that[3] $D_1 \sqsubseteq D_2 \Leftrightarrow D_1 + D_2 = D_2$ and that the *lub* operation on $(\mathbb{C}, \sqsubseteq)$ coincides with $\sum$.

The *tree* operation maps clauses to collections. Indeed, every clause $c := p(\boldsymbol{t}) \leftarrow \boldsymbol{B}$ can be viewed as the "one step" interpretation (collection)

$$tree(c) := \phi\left[\left\{p(\boldsymbol{x}), p(\boldsymbol{x}) \xrightarrow[c]{\{\boldsymbol{x}/\boldsymbol{t}\}} \boldsymbol{B}\right\} \Big/ p(\boldsymbol{x})\right],$$

where $\boldsymbol{x}$ is a tuple of new distinct variables. Moreover, *tree* can be extended to programs simply as $tree(P) := \sum\{tree(c)\}_{c \in P}$.

Note that $\equiv_{\mathbb{C}}$ is a congruence w.r.t. $\cdot$, $\times$, $\bowtie$ and $\sum$ because of the renaming apart property and the "collecting" nature of these operations. Furthermore, given $D, D' \in \mathbb{C}$, if $D \equiv_{\mathbb{C}} D'$ then

$$\forall A \in Atoms.\, A \cdot D = A \cdot D'. \tag{6}$$

5. Operational semantics

The operational semantics of queries in the program P can be described in terms of the following transition system $\mathscr{T} := (\mathbb{C}, \overset{P}{\longmapsto})$. Since we want the rules of $\mathscr{T}$ to depend on properties of well-formed sets, rather than on the structure of a *single* derivation step, we can use the rule

$$\frac{D \in \mathbb{C}, \quad D \neq D \bowtie su(tree(P))}{D \overset{P}{\longmapsto} D \bowtie su(tree(P))}, \tag{7}$$

where, for any pure collection D,

$$su(D) := \sum\{(A \cdot D) \times Id_{\mathbb{C}}\}_{A \in Atoms}. \tag{8}$$

Note that $su(D)$ can be viewed as the *sequential unfolding* of the pure collection D and it is closed under renaming and under instantiation, since we consider all the possible evaluations of D. Note that we use the construction $\cdot \times Id_{\mathbb{C}}$ to allow the construction $\cdot \bowtie su(tree(P))$ to extend all derivations in the range of D whose last goal leftmost atom matches the head of a clause in P.

The *initial states* of $\mathscr{T}$ are all the collections of *SLD*-derivations of length zero, while the *final states* are the collections of *SLD*-refutations and finite failures.

As the intuition suggests, the transition system $\mathscr{T}$ defines the usual notion of *SLD*-derivation. The formal statement is given in Theorem 5. The specificity of this transition system is due to the fact that *we have defined it using the same semantic operators used in the denotational definition.*

[3] Remember that $+$ is an infix notation for Σ when applied to finite classes, e.g. $D_1 + D_2 := \Sigma\{D_1, D_2\}$.

Since we are interested in all the *SLD*-derivations of a query $\boldsymbol{G}$ *in* P, we define its *behavior* as

$$\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!] := \sum\{D \mid \phi_{\boldsymbol{G}} \overset{P}{\longmapsto}{}^{*} D\}, \tag{9}$$

where $\overset{P}{\longmapsto}{}^{*}$ is the reflexive and transitive closure of $\overset{P}{\longmapsto}$. The behavior (modulo variance) of a query is the operational semantics of the query. As we will see in the following (Corollary 23) every query has equivalent operational and denotational semantics.

Theorem 5. *Let P be a program and $\boldsymbol{G}$ be a goal. Then*

$$\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!] = \phi\left[\left\{\mathsf{d} \mid \mathsf{d} := \boldsymbol{G} \overset{\vartheta}{\underset{P}{\longrightarrow}}{}^{*} \boldsymbol{B}\right\} \Big/ \boldsymbol{G}\right].$$

Proof. To prove the theorem it is sufficient to show that, for any collection D and any goal $\boldsymbol{G}$, such that $D \overset{P}{\longmapsto} D \bowtie su(tree(P))$, $\bar{\mathsf{d}} \in (D \bowtie su(tree(P)))(\boldsymbol{G}) \backslash D(\boldsymbol{G})$ if and only if $\bar{\mathsf{d}} = \boldsymbol{G} \overset{\vartheta_1}{\underset{c_1}{\longrightarrow}} \cdots \overset{\vartheta_n}{\underset{c_n}{\longrightarrow}} (A, \boldsymbol{G}') \overset{\sigma}{\underset{c'}{\longrightarrow}} (\boldsymbol{B}', \boldsymbol{G}')\sigma$, where $\mathsf{d} = \boldsymbol{G} \overset{\vartheta_1}{\underset{c_1}{\longrightarrow}} \cdots \overset{\vartheta_n}{\underset{c_n}{\longrightarrow}} (A, \boldsymbol{G}') \in D(\boldsymbol{G})$, $c' = p(\boldsymbol{t}') \leftarrow \boldsymbol{B}'$ is a renamed apart (w.r.t. d) version of a clause $c \in P$ and $\sigma = mgu(A, p(\boldsymbol{t}'))$. The proof follows then by definition of $\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!]$, by definition of derivation and by a straightforward inductive argument. We prove the two implications separately.

(*Only if*) By definition of su, $\bowtie$ and since $su(D)$ is closed under renaming, there exists a derivation $\mathsf{d} = \boldsymbol{G} \overset{\vartheta_1}{\underset{c_1}{\longrightarrow}} \cdots \overset{\vartheta_n}{\underset{c_n}{\longrightarrow}} (A, \boldsymbol{G}') \in D$ and there exists a non-null derivation $\mathsf{d}_1 \in ((A \cdot tree(P)) \times \phi_{\boldsymbol{G}'})(A, \boldsymbol{G}')$ such that

$$\bar{\mathsf{d}} = \mathsf{d} :: \mathsf{d}_1 \quad \text{and} \quad var(\mathsf{d}) \cap var(\mathsf{d}_1) = var(A, \boldsymbol{G}'). \tag{10}$$

By definition of $\times$, there exists

$$\mathsf{d}_2 \in (A \cdot tree(P))(A) \backslash \{A\} \text{ such that } \mathsf{d}_1 = \mathsf{d}_2 \wedge \boldsymbol{G}'. \tag{11}$$

Moreover, by definition of $\cdot$ and $tree$, $\mathsf{d}_2 = \partial_\gamma(\mathsf{d}_c)$, where $\mathsf{d}_c = p(\boldsymbol{x}) \overset{\{\boldsymbol{x}/\boldsymbol{t}\rho\}}{\underset{c\rho}{\longrightarrow}} \boldsymbol{B}\rho$, $c = p(\boldsymbol{t}) \leftarrow \boldsymbol{B} \in P$, $\boldsymbol{x}$ is a tuple of new distinct variables, $var(A) \cap var(c\rho) = \emptyset$ and γ is an idempotent substitution such that $A = p(\boldsymbol{x})\gamma$. Then, by definition of ∂ and since $\mathsf{d}_2 \neq A$, $\mathsf{d}_2 = A \overset{\sigma}{\underset{c\rho}{\longrightarrow}} \boldsymbol{B}\rho\sigma$, where $\sigma = mgu(A, p(\boldsymbol{t})\rho)$. Moreover, by (10) and (11), $var(\mathsf{d}) \cap var(c\rho) = \emptyset$. Then, by (11) and by definition of $\wedge$, $\mathsf{d}_1 = (A, \boldsymbol{G}') \overset{\sigma}{\underset{c\rho}{\longrightarrow}} \boldsymbol{B}\rho\sigma, \boldsymbol{G}'\sigma$. Hence, $\bar{\mathsf{d}} = \boldsymbol{G} \overset{\vartheta_1}{\underset{c_1}{\longrightarrow}} \cdots \overset{\vartheta_n}{\underset{c_n}{\longrightarrow}} (A, \boldsymbol{G}') \overset{\sigma}{\underset{c\rho}{\longrightarrow}} (\boldsymbol{B}\rho, \boldsymbol{G}')\sigma$. Now, to complete it is sufficient to observe that $c\rho$ is a renamed apart (w.r.t. d) version of $c \in P$.

(*If*) By definition of derivation, $var(c') \cap var(A) = \emptyset$. Then, by definition of ∂, there exists $\mathsf{d}_1 = \partial_\gamma(p(\boldsymbol{x}) \overset{\{\boldsymbol{x}/\boldsymbol{t}'\}}{\underset{c'}{\longrightarrow}} \boldsymbol{B}')$, where $\boldsymbol{x}$ is a tuple of new distinct variables and γ is an idempotent substitution such that $A = p(\boldsymbol{x})\gamma$. Moreover, by definition of $tree$ and $\cdot$, $\mathsf{d}_1 \in (A \cdot tree(c))(A)$. Therefore, since $c \in P$, by definition of $tree$ and since

is monotonic, $\mathsf{d}_1 \in (A \cdot tree(P))(A)$. Then, by definition of $\times$, $\mathsf{d}' = \mathsf{d}_1 \wedge \boldsymbol{G}' = (A, \boldsymbol{G}') \xrightarrow[c']{\sigma} (\boldsymbol{B}'\sigma, \boldsymbol{G}'\sigma) \in ((A \cdot tree(P)) \times Id_{\mathbb{C}})(A, \boldsymbol{G}') = su(tree(P))(A, \boldsymbol{G}')$. Moreover, by definition of derivation, $var(\mathsf{d}) \cap var(\mathsf{d}') = var(A, \boldsymbol{G}')$. Then to prove the theorem it is sufficient to observe that, by definition of $\bowtie$, since $\mathsf{d} \in D(\boldsymbol{G})$ and $last(\mathsf{d}) = first(\mathsf{d}')$, $\bar{\mathsf{d}} = \mathsf{d} :: \mathsf{d}' \in (D \bowtie su(tree(P)))(\boldsymbol{G})$. □

6. The program denotation

From the notion of query behavior, we can define the behavior of a program as the collection $\sum\{\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!]\}_{G \in Goals}$. This collection can be viewed as a program denotation but we can define a better top-down program denotation. This can be obtained[4] by collecting *only* the behaviors for all pure atomic goals, i.e., the behaviors of the procedures with no constraints on the inputs. This yields a compact denotation which is a finite-domain function (that may give infinite results).

The *top-down SLD-derivations denotation* of a program P is the interpretation

$$\mathscr{O}[\![P]\!] := \sum\{\mathscr{B}[\![p(\boldsymbol{x}) \text{ in } P]\!]/_{\equiv_{\mathbb{C}}}\}_{p(\boldsymbol{x}) \in Goals}. \tag{12}$$

This can be viewed as a *program* denotation, since it is the semantics of the program as a set of definite clauses (or a set of procedure definitions).

Using standard techniques it can be proved that $\mathscr{P}[\![P]\!]$ is continuous, hence we can define the *fixpoint denotation* of the program P as the interpretation $\mathscr{F}[\![P]\!] := lfp\,\mathscr{P}[\![P]\!]$. In the following, we will prove that $\mathscr{F}[\![P]\!]$ and $\mathscr{O}[\![P]\!]$ are equivalent.

Program denotations are strictly related to program equivalences. We define the equivalence $\approx$ of two programs P_1, P_2 as the equivalence of the behaviors of the two programs, i.e.,

$$P_1 \approx P_2 \iff \forall \boldsymbol{G} \in Goals.\, \mathscr{B}[\![\boldsymbol{G} \text{ in } P_1]\!] = \mathscr{B}[\![\boldsymbol{G} \text{ in } P_2]\!].$$

Now, we give two definitions to relate program equivalences to denotations. Let $\mathscr{S}[\![P]\!]$ be a program denotation and $\sim$ be a program equivalence. Then

(1) $\mathscr{S}$ is *correct* w.r.t. $\sim$ if $\mathscr{S}[\![P_1]\!] = \mathscr{S}[\![P_2]\!] \Longrightarrow P_1 \sim P_2$,

(2) $\mathscr{S}$ is *minimal* w.r.t. $\sim$ if $P_1 \sim P_2 \Longrightarrow \mathscr{S}[\![P_1]\!] = \mathscr{S}[\![P_2]\!]$.

In the following (Corollary 12), we will prove that $\mathscr{O}[\![P]\!]$ (and $\mathscr{F}[\![P]\!]$) is correct and minimal w.r.t. $\approx$.

7. Semantic properties of *SLD*-derivations

We show that the program denotation $\mathscr{O}[\![P]\!]$ has several interesting properties, which can all be viewed as compositionality properties. The first compositionality result is Theorem 11 which shows that the semantic function $\mathscr{B}$ is compositional w.r.t.

[4] Essentially because of Theorem 11.

procedure calls (atomic goals) and composition (conjunction) inside goals (*AND*-compositionality).

Lemma 6. *Let* $D_1, D_2, D_3 \in \mathbb{C}$. *Then,*
(1) $D_1 \times (D_2 \times D_3) = (D_1 \times D_2) \times D_3$
(2) *If* $D_3 \sqsubseteq D_2$ *then* $D_1 \bowtie (D_2 \bowtie D_3) = (D_1 \bowtie D_2) \bowtie D_3$.

Proof. The proof of (1) follows by (2) of Lemma 4 and by definition of $\times$. To prove (2), first of all observe that, by definition of $\bowtie$, for any $D_1, D_2, D_3 \in \mathbb{C}$, $(D_1 \bowtie D_2) \bowtie D_3 = D_1 \bowtie ((D_2 \bowtie D_3) + D_3)$. Moreover, if $D_3 \sqsubseteq D_2$ then $D_3 \sqsubseteq D_2 \bowtie D_3$. Then, since $+$ is the *lub* operation on $\mathbb{C}$, $(D_2 \bowtie D_3) + D_3 = D_2 \bowtie D_3$ and, therefore, $D_1 \bowtie ((D_2 \bowtie D_3) + D_3) = D_1 \bowtie (D_2 \bowtie D_3)$. □

By Lemma 6 and by a straightforward inductive argument, we have that, for any $n \geqslant 2$ and $D \in \mathbb{C}$,

$$\underbrace{D \bowtie (D \bowtie (\cdots \bowtie D))}_{n} = \underbrace{((D \bowtie D) \bowtie \cdots) \bowtie D}_{n}. \tag{13}$$

Therefore, we can omit the parentheses in such formulae. Given a pure collection D, we denote by $su_n(D)$ the collection[5]

$$\underbrace{su(D) \bowtie \cdots \bowtie su(D)}_{n}.$$

Lemma 7. $\cdot$, $\times$ *and* $\bowtie$ *distributes over sums in* $(\mathbb{C}, \sqsubseteq)$.

Proof. We have to prove that, for any $\{D_j\}_{j\in J} \subseteq \mathbb{C}$ and $D \in \mathbb{C}$, $\sum\{D_j \bowtie D\}_{j\in J} = (\sum\{D_j\}_{j\in J}) \bowtie D$ and $D \bowtie (\sum\{D_j\}_{j\in J}) = \sum\{D \bowtie D_j\}_{j\in J}$. Analogously for $\cdot$ and $\times$. The proof is straightforward by observing that $\bowtie$, $\cdot$ and $\times$ are defined by collecting the results of operations defined on single derivations. □

Since the *lub* operation on $(\mathbb{C}, \sqsubseteq)$ coincides with $\sum$, a straightforward consequence is that $\cdot$, $\times$ and $\bowtie$ are continuous (and monotonic) on $(\mathbb{C}, \sqsubseteq)$.

Lemma 8. *Let* A *be an atom,* $D \in \mathbb{P}\mathbb{C}$, $D', D'' \in \mathbb{C}$ *and* $\boldsymbol{G}$ *be a goal. Then*
(1) $A \cdot (D' \bowtie su(D)) = (A \cdot D') \bowtie su(D)$.
(2) $(D' \bowtie su(D)) \times \phi_{\boldsymbol{G}} \sqsubseteq (D' \times \phi_{\boldsymbol{G}}) \bowtie su(D)$.
(3) *If* $D' \bowtie su(D) = D'$ *then* $(D' \times D'') \bowtie su(D) = D' \times (D'' \bowtie su(D))$.

Corollary 9. *Let* P *be a program and* $\boldsymbol{G}$ *a goal. Then*
(1) $\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!] = \phi_{\boldsymbol{G}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}$.
(2) $\mathscr{O}[\![P]\!] = (Id_{\mathbb{I}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0})\,/_{\equiv_{\mathbb{C}}}$.

[5] Note that $su_1(D) := su(D)$ and we assume that $su_0(D) := \phi$.

Proof. We prove (1) and (2) separately.

(1) We use the notation $D \overset{P}{\longmapsto}^n D'$ to denote that the collection D results in the collection D' with *at most* n transition steps $\overset{P}{\longmapsto}$. First of all, note that, by definition of $\overset{P}{\longmapsto}$, given a collection D, $\sum\{D' \mid D \overset{P}{\longmapsto}^1 D'\} = D \bowtie su(tree(P))$. By Lemma 7 it follows that, for any $\{D_j\}_{j\in J} \subseteq \mathbb{C}$,

$$\begin{aligned}\sum\{\sum\{D' \mid D_j \overset{P}{\longmapsto}^1 D'\}\}_{j\in J} &= \sum\{D_j \bowtie su(tree(P))\}_{j\in J}\\ &= \sum\{D_j\}_{j\in J} \bowtie su(tree(P)).\end{aligned} \tag{14}$$

We prove (by induction on n) that $\sum\{D' \mid D \overset{P}{\longmapsto}^n D'\} = D \bowtie su_n(tree(P))$. For $n = 0$, $\sum\{D' \mid D \overset{P}{\longmapsto}^0 D'\} = D = D \bowtie su_0(tree(P))$. For $n > 0$ the following facts hold:

$$\begin{aligned}&\sum\{D' \mid D \overset{P}{\longmapsto}^n D'\} && \text{(by definition of } \overset{P}{\longmapsto}^n)\\ &= \sum\{D' \mid D \overset{P}{\longmapsto}^{n-1} D'', D'' \overset{P}{\longmapsto}^1 D'\} && \text{(by set theory)}\\ &= \sum\{\sum\{D' \mid D'' \overset{P}{\longmapsto}^1 D'\} \mid D \overset{P}{\longmapsto}^{n-1} D''\} && \text{(by (14))}\\ &= \sum\{D'' \mid D \overset{P}{\longmapsto}^{n-1} D''\} \bowtie su(tree(P)) && \text{(by inductive hypothesis)}\\ &= (D \bowtie su_{n-1}(tree(P))) \bowtie su(tree(P)) && \text{(by Lemma 6 (2))}\\ &= D \bowtie su_n(tree(P))\end{aligned}$$

Finally,

$$\begin{aligned}&\mathscr{B}[\![\boldsymbol{G} \textit{ in } P]\!] && \text{(by definition)}\\ &= \sum\{D \mid \phi_{\boldsymbol{G}} \overset{P}{\longmapsto}^* D\} && \text{(by definition of } \overset{P}{\longmapsto}^*)\\ &= \sum\{\sum\{D \mid \phi_{\boldsymbol{G}} \overset{P}{\longmapsto}^n D\}\}_{n\geqslant 0} && \text{(by previous result)}\\ &= \sum\{\phi_{\boldsymbol{G}} \bowtie su_n(tree(P))\}_{n\geqslant 0} && \text{(by Lemma 7).}\\ &= \phi_{\boldsymbol{G}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}\end{aligned}$$

(2) The following facts hold:

$$\begin{aligned}&\mathscr{O}[\![P]\!]\\ &= \sum\{\mathscr{B}[\![p(\boldsymbol{x}) \textit{ in } P]\!]/_{\equiv_{\mathbb{C}}}\}_{p(\boldsymbol{x})\in Goals} && \text{(by definition)}\\ &= \left(\sum\{\mathscr{B}[\![p(\boldsymbol{x}) \textit{ in } P]\!]\}_{p(\boldsymbol{x})\in Goals}\right)/_{\equiv_{\mathbb{C}}} && \left(\text{since } \equiv_{\mathbb{C}} \text{ is a congruence w.r.t. } \sum\right)\\ &= \left(\sum\{\phi_{p(\boldsymbol{x})} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}\}_{p(\boldsymbol{x})\in Goals}\right)/_{\equiv_{\mathbb{C}}} && \text{(by (1) above)}\\ &= \left(\sum\{\phi_{p(\boldsymbol{x})}\}_{p(\boldsymbol{x})\in Goals} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}\right)/_{\equiv_{\mathbb{C}}} && \text{(Lemma 7)}\\ &= (Id_{\mathbb{I}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0})/_{\equiv_{\mathbb{C}}} && \text{(by definition of } Id_{\mathbb{I}}). \quad \square\end{aligned}$$

The following (technical) corollary follows by Lemmas 8, 6 and a straightforward inductive argument.

Corollary 10. *Let A be an atom, $\boldsymbol{G}$ be a goal, $D \in \mathbb{PC}$ and $D', D'' \in \mathbb{C}$. Then, for any $n \geqslant 0$,*
(1) $A \cdot (D' \bowtie su_n(D)) = (A \cdot D') \bowtie su_n(D)$.
(2) $(D' \bowtie su_n(D)) \times \phi_{\boldsymbol{G}} \sqsubseteq (D' \times \phi_{\boldsymbol{G}}) \bowtie su_n(D)$.
(3) *If* $D' \bowtie su_n(D) = D'$ *then* $(D' \times D'') \bowtie su_n(D) = D' \times (D'' \bowtie su_n(D))$.

Essentially, because of Corollary 10 we can always reconstruct an *SLD*-tree for a generic (non-pure and non-atomic) goal from the *SLD*-trees of pure atoms.

Theorem 11. *Let A be an atom, $\boldsymbol{G}_1$ and $\boldsymbol{G}_2$ be goals and P be a program. Then*
(1) $\mathscr{B}[\![A \text{ in } P]\!] = A \cdot \mathscr{O}[\![P]\!]$.
(2) $\mathscr{B}[\![(\boldsymbol{G}_1, \boldsymbol{G}_2) \text{ in } P]\!] = \mathscr{B}[\![\boldsymbol{G}_1 \text{ in } P]\!] \times \mathscr{B}[\![\boldsymbol{G}_2 \text{ in } P]\!]$.

Proof. We prove (1) and (2) separately.
(1) The following equalities hold:

$$
\begin{aligned}
&\mathscr{B}[\![A \text{ in } P]\!] \\
&\quad = \phi_A \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} \quad \text{(by Corollary 9 (1))} \\
&\quad = (A \cdot Id_{\mathbb{I}}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} \quad \text{(by definition of } \cdot \text{ and } Id_{\mathbb{I}}) \\
&\quad = A \cdot (Id_{\mathbb{I}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}) \quad \text{(by Corollary 10 and Lemma 7)}
\end{aligned}
$$

Finally, since (by 6) $\equiv_{\mathbb{C}}$ is a congruence w.r.t. $\cdot$ and by Corollary 9, $A \cdot (Id_{\mathbb{I}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}) = A \cdot \mathscr{O}[\![P]\!]$.
(2) First of all, note that, for any goal $\boldsymbol{G}$ and any $n \geqslant 0$,

$$
\begin{aligned}
&\mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!] \bowtie su_n(tree(P)) \\
&\quad = (\phi_{\boldsymbol{G}} \bowtie \sum\{su_k(tree(P))\}_{k\geqslant 0}) \bowtie su_n(tree(P)) \quad \text{(by Corollary 9)} \\
&\quad = \phi_{\boldsymbol{G}} \bowtie \sum\{su_k(tree(P)) \bowtie su_n(tree(P))\}_{k\geqslant 0} \quad \text{(by Lemmata 6 and 7)} \\
&\quad = \phi_{\boldsymbol{G}} \bowtie \sum\{su_k(tree(P))\}_{k\geqslant 0} \quad \text{(by (13))} \\
&\quad = \mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!] \quad \text{(by Corollary 9)}
\end{aligned}
$$

Now, we prove the two inclusions of the theorem separately.
($\sqsubseteq$): In this case the following facts hold.

$$
\begin{aligned}
&\mathscr{B}[\![(\boldsymbol{G}_1, \boldsymbol{G}_2) \text{ in } P]\!] \\
&\quad = (\phi_{\boldsymbol{G}_1} \times \phi_{\boldsymbol{G}_2}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} \\
&\qquad \text{(by Corollary 9 (1) and by definition of } \times) \\
&\quad \sqsubseteq (\mathscr{B}[\![\boldsymbol{G}_1 \text{ in } P]\!] \times \phi_{\boldsymbol{G}_2}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} \\
&\qquad \text{(since } \times \text{ and } \bowtie \text{ are monotonic and } \phi_{\boldsymbol{G}_1} \sqsubseteq \mathscr{B}[\![\boldsymbol{G}_1 \text{ in } P]\!])
\end{aligned}
$$

$= \mathscr{B}[\![\boldsymbol{G}_1 \textit{ in } P]\!] \times (\phi_{\boldsymbol{G}_2} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0})$

(by previous observation, Corollary 10 (3) and Lemma 7)

$= \mathscr{B}[\![\boldsymbol{G}_1 \textit{ in } P]\!] \times \mathscr{B}[\![\boldsymbol{G}_2 \textit{ in } P]\!]$ (by Corollary 9).

($\sqsupseteq$): The following facts hold.

$\mathscr{B}[\![\boldsymbol{G}_1 \textit{ in } P]\!] \times \mathscr{B}[\![\boldsymbol{G}_2 \textit{ in } P]\!]$

$= (\mathscr{B}[\![\boldsymbol{G}_1 \textit{ in } P]\!] \times \phi_{\boldsymbol{G}_2}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}$

(by repeating previous steps)

$= ((\phi_{\boldsymbol{G}_1} \bowtie \sum\{su_k(tree(P))\}_{k\geqslant 0}) \times \phi_{\boldsymbol{G}_2}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}$

(by Corollary 9)

$\sqsubseteq ((\phi_{\boldsymbol{G}_1} \times \phi_{\boldsymbol{G}_2}) \bowtie \sum\{su_k(tree(P))\}_{k\geqslant 0}) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}$

(by Corollary 10(2) and Lemma 7)

$\sqsubseteq \mathscr{B}[\![(\boldsymbol{G}_1, \boldsymbol{G}_2) \textit{ in } P]\!] \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}$

(by definition of $\times$ and by Point 1 of Corollary 9)

$= \mathscr{B}[\![(\boldsymbol{G}_1, \boldsymbol{G}_2) \textit{ in } P]\!]$

(by Lemma 7 and by previous observation). □

From Theorem 11 we can immediately derive that, for any atom A, goal $\boldsymbol{G}$ and program P,

$$\mathscr{B}[\![\Box \textit{ in } P]\!] = \phi_{\Box}, \tag{15}$$

$$\mathscr{B}[\![(A, \boldsymbol{G}) \textit{ in } P]\!] = (A \cdot \mathscr{O}[\![P]\!]) \times \mathscr{B}[\![\boldsymbol{G} \textit{ in } P]\!]. \tag{16}$$

The above closure property of $\mathscr{O}$ w.r.t. $\mathscr{B}$ allows us to show that the denotation $\mathscr{O}$ is correct and minimal w.r.t. $\approx$.

Corollary 12. *Let P_1 and P_2 be two programs. Then $P_1 \approx P_2 \iff \mathscr{O}[\![P_1]\!] = \mathscr{O}[\![P_2]\!]$.*

Proof. The proof of the implication $\Longrightarrow$ is straightforward by definition of $\approx$ and of $\mathscr{O}$. The proof of the other implication is by contradiction. Assume that $P_1 \not\approx P_2$ and $\mathscr{O}[\![P_1]\!] = \mathscr{O}[\![P_2]\!]$. By definition of $\approx$, there exists $\boldsymbol{G} \in \mathit{Goals}$ such that $\mathscr{B}[\![\boldsymbol{G} \textit{ in } P_1]\!] \neq \mathscr{B}[\![\boldsymbol{G} \textit{ in } P_2]\!]$. Now the proof is by structural induction on $\boldsymbol{G}$.

$\boldsymbol{G} = \Box$: Contradictory, since by (15), $\mathscr{B}[\![\Box \textit{ in } P_1]\!] = \phi_{\Box} = \mathscr{B}[\![\Box \textit{ in } P_2]\!]$.

$\boldsymbol{G} = (A, \boldsymbol{G}')$: By (16) two cases arise. If $A \cdot \mathscr{O}[\![P_1]\!] \neq A \cdot \mathscr{O}[\![P_2]\!]$, $\mathscr{O}[\![P_1]\!] \neq \mathscr{O}[\![P_2]\!]$ and this contradicts the hypothesis.

Otherwise $\mathscr{B}[\![\boldsymbol{G}' \textit{ in } P_1]\!] \neq \mathscr{B}[\![\boldsymbol{G}' \textit{ in } P_2]\!]$ and then, by inductive hypothesis, we have a contradiction. □

Using the replacement operator we can define a semantic operator $\uplus$ which computes the *OR*-composition of two denotations. Namely, given $D_1, D_2 \in \mathbb{P}\mathbb{C}$, $D_1 \uplus D_2 := [D_1 +$

$D_2]^*$ where $[D]^*$ is the least solution of the equation $[D]^* = Id_{\mathbb{I}} + ([D]^* \bowtie su(D))$. Theorem 15 shows the *OR*-compositionality property of $\mathcal{O}[\![P]\!]$, i.e., the compositionality w.r.t. the $\cup$ operator. First, we need the following (technical) lemma.

Lemma 13. *Let D, $D' \in \mathbb{PC}$. Then $su(D \bowtie su(D')) \sqsubseteq su(D) \bowtie su(D')$.*

Proof. The following facts hold:

$$
\begin{aligned}
& su(D \bowtie su(D')) \\
& \quad = \sum\{(A \cdot (D \bowtie su(D'))) \times Id_{\mathbb{C}}\}_{A \in Atoms} \\
& \qquad \text{(by definition of } su\text{)} \\
& \quad = \sum\{((A \cdot D) \bowtie su(D')) \times \sum\{\phi_G\}_{G \in Goals}\}_{A \in Atoms} \\
& \qquad \text{(by Lemma 8(1) and by definition of } Id_{\mathbb{C}}\text{)} \\
& \quad = \sum\{\sum\{((A \cdot D) \bowtie su(D')) \times \phi_G\}_{G \in Goals}\}_{A \in Atoms} \\
& \qquad \text{(by Lemma 7)} \\
& \quad \sqsubseteq \sum\{\sum\{((A \cdot D) \times \phi_G) \bowtie su(D')\}_{G \in Goals}\}_{A \in Atoms} \\
& \qquad \text{(by Lemma 8(2))} \\
& \quad = \sum\{(A \cdot D) \times \sum\{\phi_G\}_{G \in Goals}\}_{A \in Atoms} \bowtie su(D') \\
& \qquad \text{(by Lemma 7)} \\
& \quad = su(D) \bowtie su(D') \\
& \qquad \text{(by definition of } Id_{\mathbb{C}} \text{ and of } su\text{).} \qquad \square
\end{aligned}
$$

Corollary 14. *Let $D, D' \in \mathbb{PC}$. Then, for any $k \geqslant 0$, $su(D \bowtie su_k(D')) \sqsubseteq su(D) \bowtie suk(D')$.*

Proof. If $k = 0$, by definition of $\bowtie$ and since $su_0(D') = \phi$, $su(D \bowtie \phi) = su(D) = su(D) \bowtie \phi$. Otherwise the proof is by induction on $k > 0$.

First of all note that, by definition of su_k, $su(D \bowtie su_k(D')) = su(D \bowtie (su_{k-1}(D') \bowtie su(D')))$ and, for $k > 1$, $su(D') \sqsubseteq su_{k-1}(D')$. Then by Lemma 6, $D \bowtie su_k(D') = (D \bowtie su_{k-1}(D')) \bowtie su(D')$. This result trivially holds also for $k = 1$. To conclude

$$
\begin{aligned}
& su(D \bowtie su_k(D')) \\
& \quad = su((D \bowtie su_{k-1}(D')) \bowtie su(D')) \quad \text{(by previous result)} \\
& \quad \sqsubseteq su(D \bowtie su_{k-1}(D')) \bowtie su(D') \quad \text{(by Lemma 13)} \\
& \quad \sqsubseteq (su(D) \bowtie su_{k-1}(D')) \bowtie su(D') \quad \text{(by inductive hypothesis)} \\
& \quad = su(D) \bowtie su_k(D') \quad \text{(by previous result).} \qquad \square
\end{aligned}
$$

Theorem 15. *Let P_1 and P_2 be programs. Then $\mathcal{O}[\![P_1 \cup P_2]\!] = \mathcal{O}[\![P_1]\!] \uplus \mathcal{O}[\![P_2]\!]$.*

Proof. By definition of $\uplus$, $\mathscr{O}[\![P_1]\!] \uplus \mathscr{O}[\![P_2]\!] = [\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]]^*$. Since $\mathbb{PC}$ is a complete lattice and by $[\cdot]^*$ definition, $[D]^* = lfp(\mathscr{H}_D) = \mathscr{H}_D{\uparrow}\omega = \sum\{\mathscr{H}_D{\uparrow}n\}_{n\geqslant 0}$, where $\mathscr{H}_D : \mathbb{PC} \longrightarrow \mathbb{PC}$ is the continuous function $\mathscr{H}_D(D') = Id_{\mathbb{I}} + (D' \bowtie su(D))$. First of all, we prove by induction that, for any $n \geqslant 1$, $\mathscr{H}_D{\uparrow}n = Id_{\mathbb{I}} \bowtie su_{n-1}(D)$. Then, by Lemma 7, $[D]^* = Id_{\mathbb{I}} \bowtie \sum\{su_n(D)\}_{n\geqslant 0}$.

$n = 1$: $\mathscr{H}_D{\uparrow}1 = \mathscr{H}_D(\phi) = Id_{\mathbb{I}} + (\phi \bowtie su(D)) = Id_{\mathbb{I}} = Id_{\mathbb{I}} \bowtie su_0(D)$.

$n > 1$: The following hold:

$$
\begin{aligned}
&\mathscr{H}_D{\uparrow}n \\
&\quad = \mathscr{H}_D(\mathscr{H}_D{\uparrow}n-1) \quad \text{(by definition of } \uparrow n) \\
&\quad = \mathscr{H}_D(Id_{\mathbb{I}} \bowtie su_{n-1}(D)) \quad \text{(by inductive hypothesis)} \\
&\quad = Id_{\mathbb{I}} + ((Id_{\mathbb{I}} \bowtie su_{n-1}(D)) \bowtie su(D)) \quad \text{(by definition of } \mathscr{H}_D) \\
&\quad = Id_{\mathbb{I}} + (Id_{\mathbb{I}} \bowtie su_n(D)) \quad \text{(by Lemma 6)} \\
&\quad = Id_{\mathbb{I}} \bowtie su_n(D) \quad \text{(since } \bowtie \text{ is extensive).}
\end{aligned}
$$

Now, to prove the theorem, we have to prove that $\sum\{su_n(tree(P_1 \cup P_2))\}_{n\geqslant 0} = \sum\{su_n(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])\}_{n\geqslant 0}$. We prove the two inclusions separately.

($\sqsubseteq$): First of all observe that, since (for any program P) $tree(P)$ is a pure collection, $tree(P) \sqsubseteq Id_{\mathbb{I}} \bowtie su(tree(P)) \sqsubseteq \mathscr{O}[\![P]\!]$. Then, by definition of $tree$, $tree(P_1 \cup P_2) = tree(P_1) + tree(P_2) \sqsubseteq \mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]$ and therefore, since $\cdot$ and $\times$ are monotonic, $su(tree(P_1 \cup P_2)) \sqsubseteq su(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])$. Then, since $\bowtie$ is also monotonic, for any $n \geqslant 0$, $su_n(tree(P_1 \cup P_2)) \sqsubseteq su_n(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])$.

($\sqsupseteq$): We prove (by induction on h) that, for any derivation d, if there exists $h \geqslant 0$ such that $\mathsf{d} \in su_h(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])(\boldsymbol{G})$ then there exists $k \geqslant 0$ such that $\mathsf{d} \in su_k(tree(P_1 \cup P_2))(\boldsymbol{G})$. If $h = 0$ simply choose $k = 0$. Otherwise let $h > 0$ and observe that, by definition of su_h and by Lemma 6,

$$
su_h(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]) = su_{h-1}(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]) \bowtie su(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]). \tag{17}
$$

We have two possibilities. If $\mathsf{d} \in su_{h-1}(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])(\boldsymbol{G})$ then, by inductive hypothesis, there exists $k \geqslant 0$ such that $\mathsf{d} \in su_k(tree(P_1 \cup P_2))(\boldsymbol{G})$ and then the theorem follows.

Otherwise, by definition of $\bowtie$, by (17) and since $su(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])$ is closed under renaming, $\mathsf{d} = \mathsf{d}_1 :: \mathsf{d}_2$, where $\mathsf{d}_1 \in su_{h-1}(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!])(\boldsymbol{G})$, $last(\mathsf{d}_1) = \boldsymbol{B}$ and

$$
\mathsf{d}_2 \in (su(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]))(\boldsymbol{B}). \tag{18}
$$

By inductive hypothesis, there exists $m \geqslant 0$ such that

$$
\mathsf{d}_1 \in su_m(tree(P_1 \cup P_2))(\boldsymbol{G}). \tag{19}
$$

Now note that, by Lemma 7, $su(\mathscr{O}[\![P_1]\!] + \mathscr{O}[\![P_2]\!]) = su(\mathscr{O}[\![P_1]\!]) + su(\mathscr{O}[\![P_2]\!])$ and therefore, by (18), $\mathsf{d}_2 \in (su(\mathscr{O}[\![P_1]\!]) + su(\mathscr{O}[\![P_2]\!]))(\boldsymbol{B})$. Now assume, without loss of generality, that $\mathsf{d}_2 \in su(\mathscr{O}[\![P_1]\!])(\boldsymbol{B})$. Then, by (2) of Corollary 9, by Lemma 7 and since $tree(P_1) \sqsubseteq$

$tree(P_1 \cup P_2)$, there exists $l \geqslant 0$ such that

$$\mathsf{d}_2 \in su(Id_{\mathbb{I}} \bowtie su_l(tree(P_1)))(\boldsymbol{B}) \subseteq su(Id_{\mathbb{I}} \bowtie su_l(tree(P_1 \cup P_2)))(\boldsymbol{B}). \tag{20}$$

Moreover, by Corollary 14 and since $su(Id_{\mathbb{I}}) \sqsubseteq Id_{\mathbb{C}}$, $su(Id_{\mathbb{I}} \bowtie su_l(tree(P_1 \cup P_2))) \sqsubseteq su(Id_{\mathbb{I}}) \bowtie su_l(tree(P_1 \cup P_2)) \sqsubseteq Id_{\mathbb{C}} \bowtie su_l(tree(P_1 \cup P_2))$.

By the previous result and by (20), $\mathsf{d}_2 \in (Id_{\mathbb{C}} \bowtie su_l(tree(P_1 \cup P_2)))(\boldsymbol{B})$ and therefore, by (19) and since $\mathsf{d} = \mathsf{d}_1 :: \mathsf{d}_2$, $\mathsf{d} \in (su_m(tree(P_1 \cup P_2)) \bowtie (Id_{\mathbb{C}} \bowtie su_l(tree(P_1 \cup P_2))))(\boldsymbol{G})$. Finally, note that, by definition of $\bowtie$, for any $D \in \mathbb{C}$, $Id_{\mathbb{C}} \bowtie D = Id_{\mathbb{C}} + D$ and $D \bowtie Id_{\mathbb{C}} = D$. Then, by (13) and since $\bowtie$ is additive and extensive, $\mathsf{d} \in (su_m(tree(P_1 \cup P_2)) \bowtie su_l(tree(P_1 \cup P_2)))(\boldsymbol{G}) = su_{m+l}(tree(P_1 \cup P_2))(\boldsymbol{G})$. □

In Theorem 21 we will prove that the top-down and the bottom-up denotations are indeed equivalent, which implies (by Theorem 11) the equivalence between the denotational and the operational semantics.

In the following, to simplify the notation, given a pure collection D, we denote by $pu(D)$, $pu_n(D)$ and $pu^n(D)$, respectively, the collections[6]

$$pu(D) := \sum\{\mathscr{G}[\![\boldsymbol{G}]\!]_D\}_{\boldsymbol{G} \in Goals}, \tag{21}$$

$$pu_n(D) := \underbrace{pu(D) \bowtie \cdots \bowtie pu(D)}_{n}, \tag{22}$$

$$pu^n(D) := \underbrace{pu(D \bowtie pu(D \bowtie pu(\cdots)))}_{n}. \tag{23}$$

Note that $pu(D)$ can be viewed as the *parallel unfolding* of the pure collection D and (analogously to $su(D)$) it is closed under renaming and under instantiation, since we consider all the possible evaluations of D. It is interesting to note that the operators su and pu enjoy some closure properties. Namely, given a pure collection D the following properties hold.

- If d is a renamed version of an element $\mathsf{d}' \in su(D)(\boldsymbol{G})$, by using a renaming ρ, then $\mathsf{d} \in su(D)(\boldsymbol{G}\rho)$. The same holds for pu.
- Using Lemma 4, it is easy to check that, for any idempotent substitution γ, goal $\boldsymbol{G}$ and derivation d, such that $\partial_\gamma(\mathsf{d})$ is defined, $\mathsf{d} \in su(D)(\boldsymbol{G})$ implies $\partial_\gamma(\mathsf{d}) \in su(D)(\boldsymbol{G}\gamma)$. Moreover, if $\mathsf{d} \in su(D)(\boldsymbol{G}\gamma)$ and $var(\mathsf{d}) \cap var(\boldsymbol{G}) \subseteq var(\boldsymbol{G}\gamma)$, there exists a derivation $\mathsf{d}' \in su(D)(\boldsymbol{G})$ such that $clauses(\mathsf{d}') = clauses(\mathsf{d})$ and $\mathsf{d} = \partial_\gamma(\mathsf{d}')$. The same for holds pu.

Using the definition of $pu(D)$ we can replace the definition (5) of $\mathscr{C}$ by the equation $\mathscr{C}[\![c]\!]_I = tree(c) \bowtie pu(I)$ and it is easy to check that

$$\mathscr{P}[\![P]\!]_I = Id_{\mathbb{I}} + (tree(P) \bowtie pu(I)). \tag{24}$$

The proof of the equivalence between the denotational and the operational semantics is mainly achieved by proving that the parallel unfolding can be simulated by the

[6] Note that $pu_1(D) := pu^1(D) := pu(D)$ and we assume that $pu_0(D) := pu^0(D) := \phi$.

sequential one. Corollary 17 proves a form of associativity of the parallel unfolding which reverses from bottom-up to top-down. Lemma 18 then states that a step of sequential unfolding can be safely replaced by a step of parallel unfolding and that the parallel unfolding of a (finite) goal can be simulated by (a finite number of steps of) the sequential unfolding.

Lemma 16. *Let $D, D' \in \mathbb{PC}$ and $h \geqslant 0$. Then*
(1) $pu(D) \bowtie pu(D') \sqsubseteq pu(D \bowtie pu(D'))$.
(2) $pu^h(D) \sqsubseteq \sum\{pu_k(D)\}_{k \geqslant 0}$.

Corollary 17. *Let $D \in \mathbb{PC}$. Then $\sum\{pu^h(D)\}_{h \geqslant 0} = \sum\{pu_k(D)\}_{k \geqslant 0}$.*

Proof. The inclusion $\sqsubseteq$ is straightforward by Lemma 16. For the other inclusion we prove (by induction on n) that, for any $n \geqslant 0$, $pu_n(D) \sqsubseteq pu^n(D)$. If $n = 0$, by definition, $pu^0(D) = \phi = pu_0(D)$. Otherwise the following hold:

$$
\begin{aligned}
& pu_n(D) \\
& \quad = pu(D) \bowtie pu_{n-1}(D) && \text{(by (22) and (13))} \\
& \quad \sqsubseteq pu(D) \bowtie pu^{n-1}(D) && \text{(by inductive hypothesis and since } \bowtie \text{ is monotonic)} \\
& \quad \sqsubseteq pu(D \bowtie pu^{n-1}(D)) && \text{(by Lemma 16)} \\
& \quad = pu^n(D) && \text{(by (23)).} \quad \square
\end{aligned}
$$

Lemma 18. *Let $D \in \mathbb{PC}$. Then*
(1) $su(D) \sqsubseteq pu(D + Id_{\mathbb{I}})$.
(2) $pu(D) \sqsubseteq Id_{\mathbb{C}} + \sum\{su_n(D)\}_{n \geqslant 0}$.

Corollary 19. *Let $D \in \mathbb{PC}$. Then*

$$\sum\{pu^h(Id_{\mathbb{I}} + D)\}_{h \geqslant 0} = Id_{\mathbb{C}} + \sum\{su_k(D)\}_{k \geqslant 0}.$$

Proof. The inclusion $\sqsupseteq$ is straightforward by Corollary 17 and by (1) of Lemma 18. For the other inclusion we prove (by induction on h) that, for any $h \geqslant 0$, $pu_h(Id_{\mathbb{I}} + D) \sqsubseteq Id_{\mathbb{C}} + \sum\{su_k(D)\}_{k \geqslant 0}$. Then the hypothesis follows by Corollary 17.

$h = 0$: Straightforward, since $pu_0(Id_{\mathbb{I}} + D) = \phi$.

$h = 1$: Straightforward by (2) of Lemma 18.

$h > 1$: Let $\boldsymbol{G}$ be a goal and $\mathsf{d} \in pu_h(Id_{\mathbb{I}} + D)(\boldsymbol{G})$. By (22) and (2) of Lemma 6, $\mathsf{d} \in (pu_{h-1}(Id_{\mathbb{I}} + D) \bowtie pu(Id_{\mathbb{I}} + D))(\boldsymbol{G})$. If $\mathsf{d} \in pu_{h-1}(Id_{\mathbb{I}} + D)(\boldsymbol{G})$ then the thesis follows by inductive hypothesis (and definition of $\sqsubseteq$). Otherwise, by definition of $\bowtie$, we can assume that

$$
\begin{aligned}
& \mathsf{d} = \mathsf{d}_1 :: \mathsf{d}_2, \quad \text{where } \mathsf{d}_1 \in pu_{h-1}(Id_{\mathbb{I}} + D)(\boldsymbol{G}), \\
& last(\mathsf{d}_1) = \boldsymbol{B} \neq \square \quad \text{and} \quad \mathsf{d}_2 \in pu(Id_{\mathbb{I}} + D)(\boldsymbol{B}). \qquad (25)
\end{aligned}
$$

By inductive hypothesis, $\mathsf{d}_1 \in (Id_{\mathbb{C}} + \sum\{su_k(D)\}_{k\geqslant 0})(\boldsymbol{G})$ and then there exists $n \geqslant 0$ such that

$$\mathsf{d}_1 \in (Id_{\mathbb{C}} + su_n(D))(\boldsymbol{G}). \tag{26}$$

Moreover, by (2) of Lemma 18 and by (25), $\mathsf{d}_2 \in (Id_{\mathbb{C}} + (\sum\{su_k(Id_{\mathbb{I}} + D))\}_{k\geqslant 0})(\boldsymbol{B})$ and, therefore, there exists $m \geqslant 0$ such that

$$\mathsf{d}_2 \in (Id_{\mathbb{C}} + (su_m(Id_{\mathbb{I}} + D)))(\boldsymbol{B}). \tag{27}$$

Now observe that, by Lemma 7 and since $Id_{\mathbb{C}} = su(Id_{\mathbb{I}}) + \phi_{\square}$, $su(Id_{\mathbb{I}} + D) = su(Id_{\mathbb{I}}) + su(D) = Id_{\mathbb{C}} + su(D)$. Then (since $\forall D \in \mathbb{C}.D \bowtie Id_{\mathbb{C}} = D$, $Id_{\mathbb{C}} \bowtie D = Id_{\mathbb{C}} + D$) $Id_{\mathbb{C}} + (su_m(Id_{\mathbb{I}} + D)) = Id_{\mathbb{C}} + \sum\{su_k(D)\}_{k\leqslant m} + \sum\{Id_{\mathbb{C}} + su_k(D)\}_{k<m}$. Then (since $\forall D \in \mathbb{C}, k \leqslant k'.su_k(D) \sqsubseteq su_{k'}(D)$) we obtain $Id_{\mathbb{C}} + (su_m(Id_{\mathbb{I}} + D)) = Id_{\mathbb{C}} + su_m(D)$ and therefore, by (27), $\mathsf{d}_2 \in (Id_{\mathbb{C}} + su_m(D))(\boldsymbol{B})$. Then the following hold:

$$\begin{aligned}
&\mathsf{d}\\
&\in ((Id_{\mathbb{C}} + su_n(D)) \bowtie (Id_{\mathbb{C}} + su_m(D)))(\boldsymbol{G})\\
&\quad \text{(by (25), by definition of } \bowtie, \text{ by (26) and last result)}\\
&= (Id_{\mathbb{C}} + (su_n(D) \bowtie Id_{\mathbb{C}}) + (su_n(D) \bowtie su_m(D)))(\boldsymbol{G})\\
&\quad \text{(by Lemma 7 and since } \bowtie \text{ is extensive)}\\
&= (Id_{\mathbb{C}} + su_n(D) \bowtie su_m(D))(\boldsymbol{G})\\
&\quad \text{(since } \forall D \in \mathbb{C}.D \bowtie Id_{\mathbb{C}} = D \text{ and since } \bowtie \text{ is extensive)}\\
&= (Id_{\mathbb{C}} + su_{n+m}(D))(\boldsymbol{G})\\
&\quad \text{(by (13)}\\
&\subseteq (Id_{\mathbb{C}} + \sum\{su_k(D)\}_{k\geqslant 0})(\boldsymbol{G})\\
&\quad \text{(by definition of } \sum). \qquad \square
\end{aligned}$$

Note that, by (21), by (8) and by definition of $\bowtie$, $+$ and $Id_{\mathbb{I}}$, for any $D \in \mathbb{PC}$,

$$Id_{\mathbb{I}} + D = Id_{\mathbb{I}} \bowtie pu(D) = Id_{\mathbb{I}} \bowtie su(D). \tag{28}$$

Corollary 20. *For any program P, $\mathscr{F}[\![P]\!] = (Id_{\mathbb{I}} + tree(P)) \bowtie \sum\{pu^n(Id_{\mathbb{I}} + tree(P))\}_{n\geqslant 0}$.*

Proof. First of all recall that $\mathscr{P}[\![P]\!]$ is continuous and ϕ is the bottom of $\mathbb{C}$. We will prove (by induction on n) that, for any $n > 0$, $\mathscr{P}[\![P]\!]\uparrow n = (Id_{\mathbb{I}} + tree(P)) \bowtie pu^{n-1}(Id_{\mathbb{I}} + tree(P))$. Then, by definition of $\mathscr{F}[\![P]\!]$, $\mathscr{F}[\![P]\!] = \sum\{\mathscr{P}[\![P]\!]\uparrow n\}_{n\geqslant 0} = \sum\{(Id_{\mathbb{I}} + tree(P)) \bowtie pu^n(Id_{\mathbb{I}} + tree(P))\}_{n\geqslant 0}$ and then the thesis follows by Lemma 7.

$n = 1$: By (24), $\mathscr{P}[\![P]\!]\uparrow 1 = Id_{\mathbb{I}} + (tree(P) \bowtie pu(\phi)) = Id_{\mathbb{I}} + tree(P) = (Id_{\mathbb{I}} + tree(P)) \bowtie pu^0(Id_{\mathbb{I}} + tree(P))$.

$n > 1$: The following hold:

$$
\begin{aligned}
&\mathscr{P}[\![P]\!]\uparrow n \\
&\quad = \mathscr{P}[\![P]\!]\uparrow n + \mathscr{P}[\![P]\!]\uparrow n - 1 \\
&\qquad \text{(since } \mathscr{P}[\![P]\!]\uparrow n - 1 \sqsubseteq \mathscr{P}[\![P]\!]\uparrow n\text{)} \\
&\quad = (Id_{\mathbb{I}} + (tree(P) \bowtie pu(\mathscr{P}[\![P]\!]\uparrow n - 1))) + \mathscr{P}[\![P]\!]\uparrow n - 1 \\
&\qquad \text{(by definition of } \uparrow n \text{ and (24))} \\
&\quad = (tree(P) \bowtie pu(\mathscr{P}[\![P]\!]\uparrow n - 1)) + (Id_{\mathbb{I}} \bowtie pu(\mathscr{P}[\![P]\!]\uparrow n - 1)) \\
&\qquad \text{(by (28))} \\
&\quad = (Id_{\mathbb{I}} + tree(P)) \bowtie pu(\mathscr{P}[\![P]\!]\uparrow n - 1) \\
&\qquad \text{(by Lemma 7)} \\
&\quad = (Id_{\mathbb{I}} + tree(P)) \bowtie pu((Id_{\mathbb{I}} + tree(P)) \bowtie pu^{n-2}(Id_{\mathbb{I}} + tree(P))) \\
&\qquad \text{(by inductive hypothesis)} \\
&\quad = (Id_{\mathbb{I}} + tree(P)) \bowtie pu^{n-1}(Id_{\mathbb{I}} + tree(P)) \\
&\qquad \text{(by definition of } pu^n\text{).} \quad \square
\end{aligned}
$$

Theorem 21. *Let P be a program. Then $\mathscr{O}[\![P]\!] = \mathscr{F}[\![P]\!]$.*

Proof. By Corollaries 20 and 19, and since $D \bowtie Id_{\mathbb{C}} = D$,

$$
\begin{aligned}
\mathscr{F}[\![P]\!] &= (Id_{\mathbb{I}} + tree(P)) \bowtie \sum\{pu^n(Id_{\mathbb{I}} + tree(P))\}_{n\geqslant 0} \\
&= (Id_{\mathbb{I}} + tree(P)) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0}. \qquad (29)
\end{aligned}
$$

Now, since $tree(P)$ is a pure collection and by (28), $Id_{\mathbb{I}} + tree(P) = Id_{\mathbb{I}} \bowtie su(tree(P))$. Finally, by Lemma 6, by (29), by (2) of Corollary 9, and by a straightforward inductive argument, $\mathscr{F}[\![P]\!] = (Id_{\mathbb{I}} \bowtie su(tree(P))) \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} = Id_{\mathbb{I}} \bowtie \sum\{su_n(tree(P))\}_{n\geqslant 0} = \mathscr{O}[\![P]\!]$. $\square$

Now, we can show the *OR*-compositionality of the fixpoint denotation and the equivalence between the denotational and the operational semantics. The following corollary follows immediately from Theorems 21 and 15.

Corollary 22. *Let P_1, P_2 be programs. Then $\mathscr{F}[\![P_1 \cup P_2]\!] = \mathscr{F}[\![P_1]\!] \uplus \mathscr{F}[\![P_2]\!]$.*

Corollary 23. *For any goal $\boldsymbol{G}$ and program P, $\mathscr{Q}[\![\boldsymbol{G} \text{ in } P]\!] = \mathscr{B}[\![\boldsymbol{G} \text{ in } P]\!]$.*

Proof. The proof is by structural induction on $\boldsymbol{G}$.

$\boldsymbol{G} = \square$: By definition of $\mathscr{Q}$, $\mathscr{F}$ and $\mathscr{G}$ and by (15), $\mathscr{Q}[\![\square \text{ in } P]\!] = \mathscr{G}[\![\square]\!]_{\mathscr{F}[\![P]\!]} = \phi_{[]} = \mathscr{B}[\![\square \text{ in } P]\!]$.

$\boldsymbol{G} = (A, \boldsymbol{G}')$: The following equalities hold:

$$
\begin{aligned}
&\mathscr{2}[\![(A, \boldsymbol{G}')\ in\ P]\!] \\
&\quad = \mathscr{G}[\![(A, \boldsymbol{G}')]\!]_{\mathscr{F}[\![P]\!]} \quad \text{(by definition of } \mathscr{2} \text{ and of } \mathscr{F}) \\
&\quad = \mathscr{A}[\![A]\!]_{\mathscr{F}[\![P]\!]} \times \mathscr{2}[\![\boldsymbol{G}'\ in\ P]\!] \quad \text{(by definition of } \mathscr{G} \text{ and } \mathscr{2}) \\
&\quad = \mathscr{A}[\![A]\!]_{\mathscr{F}[\![P]\!]} \times \mathscr{B}[\![\boldsymbol{G}'\ in\ P]\!] \quad \text{(by inductive hypothesis)} \\
&\quad = (A \cdot \mathscr{F}[\![P]\!]) \times \mathscr{B}[\![\boldsymbol{G}'\ in\ P]\!] \quad \text{(by definition of } \mathscr{A}) \\
&\quad = \mathscr{B}[\![(A, \boldsymbol{G}')\ in\ P]\!] \quad \text{(by Theorem 21 and by (16)).} \qquad \square
\end{aligned}
$$

8. Conclusions and future work

As already mentioned in the introduction, our *SLD*-derivation semantics was defined as the collecting semantics of a framework for the systematic derivation of more abstract semantics, using the formal tools of abstract interpretation. The abstraction framework is described in [3] and, in more detail, in [4]. Due to the relation between the properties of the primitive semantic operators and the properties of the semantics, we can define a taxonomy of observables (abstractions). Each class in the taxonomy is characterized by a set of properties relating the primitive semantics operators and the Galois insertion which defines the observable. For each class we have

- a methodology to automatically derive the "best" abstract semantics (transition system, denotational semantics or both),
- the validity for the abstract semantics of some of the theorems which hold for the collecting semantics (equivalence between operational and denotational semantics, equivalence between top-down and bottom-up denotation, correctness, minimality and *AND* and *OR* compositionality).

The new relevant issue which can be discussed in the abstraction framework is *precision*, i.e., how good is the abstract semantics w.r.t. the abstraction of the collecting semantics. We have therefore classes of precise observables, where we can reconstruct all the semantics discussed in [2], and classes of approximate observables, where we can reconstruct several domains proposed for program analysis (groundness, types, ...). The abstraction framework has also been used as the semantic foundation of abstract diagnosis [5].

Let us finally note that, since our framework is based on standard operational and denotational semantic definitions, it can be adapted to other programming languages (especially extensions of logic programming).

Appendix A. Technical proofs

Throughout the appendix we need some technical results about properties of substitutions. Given a set of equations $E := \{s_1 = t_1, \ldots, s_n = t_n\}$, a (most general) unifier of

E is a (most general) unifier of $(s_1,\ldots,s_n)$ and $(t_1,\ldots,t_n)$. An unifiable set of equations (terms) has an idempotent *mgu*. Well-known results on idempotent *mgus* state that, if ϑ is an idempotent *mgu* of a set of equations E, then ϑ is a relevant unifier of E, i.e., $var(\vartheta) \subseteq var(E)$.

The lattice structure on idempotent substitutions [8] is isomorphic to the lattice structure on equations introduced in [16]. Therefore, we can indifferently use equations or idempotent *mgus*. The following results show the connections between the two notions that we will use in the following. Given a substitution $\vartheta := \{x_1/t_1,\ldots,x_n/t_n\}$ we define $\mathcal{E}(\vartheta) := \{x_1 = t_1,\ldots,x_n = t_n\}$. Observe that, for any substitution ϑ, $\vartheta = mgu(\mathcal{E}(\vartheta))$.

Lemma 24 (Bossi et al. [2]). *Let E_1, E_2 be sets of equations. There exists $\beta := mgu(E_1 \cup E_2)$ if and only if there exist $\vartheta := mgu(E_1)$ and $\gamma := mgu(E_2\vartheta)$ where $\beta = \vartheta\gamma$.*

Lemma 25. *Let d_1 and d_2 be derivations and γ, δ be idempotent substitutions. Then the following hold:*

(1) *If $\gamma\delta$ is idempotent and $\partial_{\gamma\delta}(\mathsf{d}_1)$, $\partial_\delta(\partial_\gamma(\mathsf{d}_1))$ are defined, then $\partial_{\gamma\delta}(\mathsf{d}_1) = \partial_\delta(\partial_\gamma(\mathsf{d}_1))$.*

(2) *If $\partial_\gamma(\mathsf{d}_1 :: \mathsf{d}_2)$ is defined, then either*
- *$length(\partial_\gamma(\mathsf{d}_1)) < length(\mathsf{d}_1)$ and $\partial_\gamma(\mathsf{d}_1 :: \mathsf{d}_2) = \partial_\gamma(\mathsf{d}_1)$ or*
- *$length(\partial_\gamma(\mathsf{d}_1)) = length(\mathsf{d}_1)$ and $\partial_\gamma(\mathsf{d}_1 :: \mathsf{d}_2) = \partial_\gamma(\mathsf{d}_1) :: \partial_{\gamma'}(\mathsf{d}_2)$, where γ' is an idempotent substitution such that $last(\partial_\gamma(\mathsf{d}_1)) = (last(\mathsf{d}_1))\gamma'$.*

Proof. The proof of (1) is straightforward by definition of ∂. To prove (2) observe that if $length(\partial_\gamma(\mathsf{d}_1)) < length(\mathsf{d}_1)$ then the proof is straightforward by definition of ∂ operation. Otherwise, let $\boldsymbol{G} := (A, \boldsymbol{G}')$, $c := H \leftarrow \boldsymbol{B}$ and γ be an idempotent substitution such that $var(c) \cap var(\gamma) = \emptyset$. Moreover, assume that $\vartheta' = mgu(A, H)$ and $\vartheta = mgu(A\gamma, H)$. We prove that there exists an idempotent substitution γ' such that $(\boldsymbol{B}, \boldsymbol{G}')\vartheta'\gamma' = (\boldsymbol{B}, \boldsymbol{G}')\gamma\vartheta$. Then the proof follows by definition of derivation and by a straightforward inductive argument.

First of all, observe that, since $var(c) \cap var(\gamma) = \emptyset$, then $\vartheta = mgu(A\gamma, H\gamma)$ and therefore, by Lemma 24,

$$\gamma\vartheta = mgu(\mathscr{E}(\gamma) \cup \{A = H\}). \tag{A.1}$$

Then, by Lemma 24, there exist $\vartheta'' = mgu(A, H)$ and $\gamma'' = mgu(\mathscr{E}(\gamma)\vartheta'')$ such that $\gamma\vartheta = \vartheta''\gamma''$. Moreover, by definition of *mgu* and since $\vartheta' = mgu(A, H)$, there exists a renaming ρ such that $\vartheta'' = \vartheta'\rho$ and therefore, by (A.1), $\gamma\vartheta = \vartheta'\rho\gamma''$. Now let $\gamma' := (\rho\gamma'')|_{(\boldsymbol{B},\boldsymbol{G}')\vartheta'}$. Then

$$\begin{aligned}
&(\boldsymbol{B}, \boldsymbol{G}')\vartheta'\gamma' \\
&\quad = (\boldsymbol{B}, \boldsymbol{G}')\vartheta'(\rho\gamma'')|_{(\boldsymbol{B},\boldsymbol{G}')\vartheta'} \quad \text{(by definition of } \gamma') \\
&\quad = (\boldsymbol{B}, \boldsymbol{G}')\vartheta'\rho\gamma'' \quad \text{(by definition of composition)} \\
&\quad = (\boldsymbol{B}, \boldsymbol{G}')\gamma\vartheta \quad \text{(since } \gamma\vartheta = \vartheta'\rho\gamma'').
\end{aligned}$$

Finally, to prove the hypothesis we have only to prove that γ' is idempotent. First of all, observe that, since ϑ' is idempotent, $dom(\vartheta') \cap var((\boldsymbol{B}, \boldsymbol{G}')\vartheta') = \emptyset$. Then, by definition of composition and since $dom(\gamma') \subseteq var((\boldsymbol{B}, \boldsymbol{G}')\vartheta')$, for any $x/t \in \gamma'$, $x/t \in \vartheta'\gamma'$ and, therefore, since $\gamma' = (\rho\gamma'')|_{(\boldsymbol{B},\boldsymbol{G}')\vartheta'}$, $x/t \in \vartheta'\rho\gamma'' = \gamma\vartheta$. Then, the hypothesis follows since by construction $\gamma\vartheta$ is an idempotent substitution. □

Now, we can give the proof of all technical lemmata.

Proof of Lemma 3. The proof of (1) and (2) is straightforward by definition of :: and ∂ operation. To prove (3) assume that $\mathsf{d}_1 = \boldsymbol{G}'_0 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \boldsymbol{G}'_k$ and let $\vartheta := \vartheta_1 \cdots \vartheta_k$, $\boldsymbol{G}''_0 := first(\mathsf{d}_2)$ such that $\mathsf{d}_1 \wedge \mathsf{d}_2$ is defined. If $\boldsymbol{G}'_k \neq \Box$ then the proof is straightforward by definition of $\wedge$. Otherwise, by definition of $\wedge$, $\mathsf{d}_1 \wedge \mathsf{d}_2 = \mathsf{d}' :: \partial_\vartheta(\mathsf{d}_2)$, where $\mathsf{d}' = (\boldsymbol{G}'_0, \boldsymbol{G}''_0) \xrightarrow[c_1]{\vartheta_1} (\boldsymbol{G}'_1, \boldsymbol{G}''_0\vartheta_1) \xrightarrow[c_2]{\vartheta_2} \cdots \xrightarrow[c_k]{\vartheta_k} \boldsymbol{G}''_0\vartheta$ is a derivation. Moreover, since $\mathsf{d}_1 \wedge \mathsf{d}_2$ is defined, $var(\mathsf{d}_1) \cap var(\mathsf{d}_2) = var(\boldsymbol{G}'_0) \cap var(\boldsymbol{G}''_0)$ and therefore $var(\mathsf{d}_1) \cap var(clauses(\mathsf{d}_2)) = \emptyset$. Then, since $var(\boldsymbol{G}''_0\vartheta) \subseteq var(\boldsymbol{G}''_0) \cup var(\mathsf{d}_1)$, $var(\boldsymbol{G}''_0\vartheta) \cap var(\mathsf{d}_2) \subseteq var(\boldsymbol{G}''_0)$ and therefore $\partial_\vartheta(\mathsf{d}_2)$ is defined and, by (2) of this lemma, $\partial_\vartheta(\mathsf{d}_2)$ is a derivation of $\boldsymbol{G}''_0\vartheta$. Finally, observe that by definition of ∂, $var(\mathsf{d}') \cap var(\partial_\vartheta(\mathsf{d}_2)) = var(\boldsymbol{G}''_0\vartheta)$. Therefore, $\mathsf{d}' :: \partial_\vartheta(\mathsf{d}_2)$ is defined and the hypothesis follows by (1). □

Proof of Lemma 4. We prove (1) and (2) separately.

(1) Let $\boldsymbol{G}_i := first(\mathsf{d}_i)$, for $i = 1, 2$. First of all, observe that, by definition of $\wedge$, it is easy to check that

$$\partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) = \partial_\gamma(\mathsf{d}_1) \wedge \boldsymbol{G}_2\gamma, \tag{A.2}$$

$$\forall \beta \in Subst.\, \partial_\beta(\mathsf{d}_2) = \partial_{\beta|_{\boldsymbol{G}_2}}(\mathsf{d}_2). \tag{A.3}$$

Then in the following we can assume, without loss of generality, that given a derivation $\partial_\beta(\mathsf{d})$, $dom(\beta) \cap var(\mathsf{d}) \subseteq var(first(\mathsf{d}))$. We distinguish the following three cases.

$last(\mathsf{d}_1) \neq \Box$: In this case $last(\partial_\gamma(\mathsf{d}_1)) \neq \Box$ and therefore, by definition of $\wedge$, $\mathsf{d}_1 \wedge \mathsf{d}_2 = \mathsf{d}_1 \wedge \boldsymbol{G}_2$ and $\partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2) = \partial_\gamma(\mathsf{d}_1) \wedge \boldsymbol{G}_2\gamma$. Then the thesis follows by (A.2).

$last(\mathsf{d}_1) = \Box$ and $last(\partial_\gamma(\mathsf{d}_1)) \neq \Box$: Let $\mathsf{d}_1 := \boldsymbol{G}_1 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \Box$ and $\vartheta := \vartheta_1 \cdots \vartheta_k$. By definition of $\wedge$,

$$\mathsf{d}_1 \wedge \mathsf{d}_2 = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2), \tag{A.4}$$

$$\partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2) = \partial_\gamma(\mathsf{d}_1) \wedge \boldsymbol{G}_2\gamma. \tag{A.5}$$

Moreover, by the previous hypothesis, $length(\partial_\gamma(\mathsf{d}_1)) < length(\mathsf{d}_1)$ and therefore $length(\partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2)) < length(\mathsf{d}_1 \wedge \boldsymbol{G}_2)$. Then, by (2) of Lemma 25, $\partial_\gamma((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2)$ and, therefore,

$$\begin{aligned}
&\partial_\gamma(\mathsf{d}_1 \wedge \mathsf{d}_2) \\
&\quad = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) \quad \text{(by (A.4) and last result)} \\
&\quad = \partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2) \quad \text{(by (A.2) and (A.5)).}
\end{aligned}$$

last$(\partial_\gamma(\mathsf{d}_1)) = \Box$: In this case *last*$(\mathsf{d}_1) = \Box$ and then there exists $k \geqslant 0$ such that $\mathsf{d}_1 = \boldsymbol{G}_1 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \Box$ and $\partial_\gamma(\mathsf{d}_1) = \boldsymbol{G}_1\gamma \xrightarrow[c_1]{\sigma_1} \cdots \xrightarrow[c_k]{\sigma_k} \Box$. Let $\vartheta := \vartheta_1 \cdots \vartheta_k$ and $\sigma := \sigma_1 \cdots \sigma_k$. Then, by definition of $\wedge$,

$$\mathsf{d}_1 \wedge \mathsf{d}_2 = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2), \tag{A.6}$$

$$\partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2) = (\partial_\gamma(\mathsf{d}_1) \wedge \boldsymbol{G}_2\gamma) :: \partial_\sigma(\partial_\gamma(\mathsf{d}_2)). \tag{A.7}$$

Now observe that, since $var(\boldsymbol{G}_1) \cap var(clauses(\mathsf{d}_1)) = \emptyset$, by definition of derivation, $\gamma\sigma$ is idempotent. Moreover, since $\mathsf{d}_1 \wedge \mathsf{d}_2$ and $\partial_\vartheta(\mathsf{d}_2)$ are defined, $(var(\gamma) \cup var(\sigma)) \cap var(clauses(\mathsf{d}_2)) = \emptyset$ and $\partial_{\sigma\vartheta}(\mathsf{d}_2)$ is defined.

Since *length*$(\partial_\gamma(\mathsf{d}_1)) =$ *length*(d_1), by (2) of Lemma 25,

$$\partial_\gamma((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\beta(\partial_\vartheta(\mathsf{d}_2)), \tag{A.8}$$

where $\boldsymbol{G}_2\gamma\sigma =$ *last*$(\partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2)) =$ *last*$(\mathsf{d}_1 \wedge \boldsymbol{G}_2)\beta = \boldsymbol{G}_2\vartheta\beta$. Then

$$(\gamma\sigma)|_{\boldsymbol{G}_2} = (\vartheta\beta)|_{\boldsymbol{G}_2} \tag{A.9}$$

and, since $\gamma\sigma$ is idempotent, $(\vartheta\beta)|_{\boldsymbol{G}_2}$ is also idempotent and, by A.5, $\partial_{\vartheta\beta}(\mathsf{d}_2)$ is defined. Finally,

$$\begin{aligned}
&\partial_\gamma(\mathsf{d}_1 \wedge \mathsf{d}_2)\\
&\quad = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\beta(\partial_\vartheta(\mathsf{d}_2)) \quad \text{(by (A.6) and (A.8)}\\
&\quad = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_{\vartheta\beta}(\mathsf{d}_2) \quad \text{(by Lemma 25(1))}\\
&\quad = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_{\gamma\sigma}(\mathsf{d}_2) \quad \text{(by (A.9) and (A.3))}\\
&\quad = \partial_\gamma(\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\sigma(\partial_\gamma(\mathsf{d}_2)) \quad \text{(by Lemma 25(1))}\\
&\quad = \partial_\gamma(\mathsf{d}_1) \wedge \partial_\gamma(\mathsf{d}_2) \quad \text{(by (A.2) and (A.7)).}
\end{aligned}$$

(2) We have two possibilities. If *last*$(\mathsf{d}_1) \neq \Box$, $\mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3) = \mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)$ and $(\mathsf{d}_1 \wedge \mathsf{d}_2) \wedge \mathsf{d}_3 = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) \wedge \boldsymbol{G}_3$. Then the proof is straightforward by definition of $\wedge$. Otherwise, let $\mathsf{d}_1 := \boldsymbol{G}_1 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \Box$ and $\vartheta := \vartheta_1 \cdots \vartheta_k$. By definition of $\wedge$,

$$\mathsf{d}_1 \wedge \mathsf{d}_2 = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2) \tag{A.10}$$

and then, by definition of $\wedge$ and (1),

$$\begin{aligned}
\mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3) &= (\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: \partial_\vartheta(\mathsf{d}_2 \wedge \mathsf{d}_3)\\
&= (\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: (\partial_\vartheta(\mathsf{d}_2) \wedge \partial_\vartheta(\mathsf{d}_3)).
\end{aligned} \tag{A.11}$$

Now two cases arise.

last$(\partial_\vartheta(\mathsf{d}_2)) \neq \Box$: In this case $\partial_\vartheta(\mathsf{d}_2) \wedge \partial_\vartheta(\mathsf{d}_3) = \partial_\vartheta(\mathsf{d}_2) \wedge \boldsymbol{G}_3\vartheta$ and, therefore,

$$\begin{aligned}
&\mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3)\\
&\quad = (\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: (\partial_\vartheta(\mathsf{d}_2) \wedge \partial_\vartheta(\mathsf{d}_3)) \quad \text{(by (A.11))}\\
&\quad = (\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: (\partial_\vartheta(\mathsf{d}_2) \wedge \boldsymbol{G}_3\vartheta) \quad \text{(by previous observation)}
\end{aligned}$$

$= ((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) \wedge \boldsymbol{G}_3$ (by definition of ::)

$= ((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) \wedge \mathsf{d}_3$ (since $last(\partial_\vartheta(\mathsf{d}_2)) \neq \Box$)

$= (\mathsf{d}_1 \wedge \mathsf{d}_2) \wedge \mathsf{d}_3$ (by (A.10)).

$last(\partial_\vartheta(\mathsf{d}_2)) = \Box$: Let $\partial_\vartheta(\mathsf{d}_2) := \boldsymbol{G}_2\vartheta \xrightarrow[c'_1]{\sigma_1} \cdots \xrightarrow[c'_m]{\sigma_m} \Box$ and $\sigma := \sigma_1 \cdots \sigma_m$. Then, by definition of $\wedge$,

$$\begin{aligned}\partial_\vartheta(\mathsf{d}_2) \wedge \partial_\vartheta(\mathsf{d}_3) &= (\partial_\vartheta(\mathsf{d}_2) \wedge \boldsymbol{G}_3\vartheta) :: \partial_\sigma(\partial_\vartheta(\mathsf{d}_3)) \\ &= \partial_\vartheta(\mathsf{d}_2 \wedge \boldsymbol{G}_3) :: \partial_\sigma(\partial_\vartheta(\mathsf{d}_3)). \end{aligned} \tag{A.12}$$

Moreover, observe that, by definition of $\wedge$,

$$\mathsf{d}_1 \wedge \mathsf{d}_2 = (\boldsymbol{G}_1, \boldsymbol{G}_2) \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \boldsymbol{G}_2\vartheta \xrightarrow[c'_1]{\sigma_1} \cdots \xrightarrow[c'_m]{\sigma_m} \Box \tag{A.13}$$

and $\vartheta\sigma$ is an idempotent substitution. Furthermore, analogously to the previous case, $\partial_{\vartheta\sigma}(\mathsf{d}_3)$ is defined. Finally,

$\mathsf{d}_1 \wedge (\mathsf{d}_2 \wedge \mathsf{d}_3)$

$= (\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: (\partial_\vartheta(\mathsf{d}_2 \wedge \boldsymbol{G}_3) :: \partial_\sigma(\partial_\vartheta(\mathsf{d}_3)))$ (by (A.11) and (A.12))

$= ((\mathsf{d}_1 \wedge (\boldsymbol{G}_2, \boldsymbol{G}_3)) :: \partial_\vartheta(\mathsf{d}_2 \wedge \boldsymbol{G}_3)) :: \partial_\sigma(\partial_\vartheta(\mathsf{d}_3))$ (since :: is associative)

$= (((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) \wedge \boldsymbol{G}_3) :: \partial_\sigma(\partial_\vartheta(\mathsf{d}_3))$ (by definition of $\wedge$)

$= (((\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: \partial_\vartheta(\mathsf{d}_2)) \wedge \boldsymbol{G}_3) :: \partial_{\vartheta\sigma}(\mathsf{d}_3)$ (by Lemma 25(1))

$= (\mathsf{d}_1 \wedge \mathsf{d}_2) \wedge \mathsf{d}_3$ (by (A.10) and (A.13)). $\square$

Proof of Lemma 8. We prove (1)–(3) separately.

(1) We prove the two inclusions separately.

($\sqsupseteq$): Let $\mathsf{d} \in ((A \cdot D') \bowtie su(D))(A)$. If $\mathsf{d} \in (A \cdot D')(A)$ then, since $\bowtie$ is extensive and $\cdot$ is monotonic, $\mathsf{d} \in (A \cdot (D' \bowtie su(D)))(A)$. Otherwise, by definition of $\bowtie$ and since $su(D)$ is closed under renaming, there exist two derivations $\mathsf{d}_1 \in (A \cdot D')(A)$ and $\mathsf{d}_2 \in su(D)(\boldsymbol{G})$ such that $\boldsymbol{G} = last(\mathsf{d}_1)$,

$$\mathsf{d} = \mathsf{d}_1 :: \mathsf{d}_2 \quad \text{and} \quad var(\mathsf{d}_1) \cap var(\mathsf{d}_2) = var(\boldsymbol{G}). \tag{A.14}$$

By definition of $\cdot$ there exists a derivation d_3, which is a renamed apart (w.r.t. A) version of an element in $D'(A')$, for some atom $A' \leqslant A$, and there exists an idempotent substitution γ such that $first(\mathsf{d}_3)\gamma = A$ and

$$\mathsf{d}_1 = \partial_\gamma(\mathsf{d}_3). \tag{A.15}$$

Without loss of generality, we can assume that

$$var(first(\mathsf{d}_3)) \cap var(\mathsf{d}_2) = \emptyset. \tag{A.16}$$

Moreover, since $D'(A')$ is a well-formed set of derivations, we can assume that

$$length(\mathsf{d}_3) = length(\mathsf{d}_1) = length(\partial_\gamma(\mathsf{d}_3)). \tag{A.17}$$

Then, by (2) of Lemma 25, there exists an idempotent substitution γ' such that $last(\mathsf{d}_1) = \boldsymbol{G} = \boldsymbol{G}'\gamma'$ where $\boldsymbol{G}' = last(\mathsf{d}_3)$. Moreover, by definition of ∂, by (A.14) and (A.16), $var(\mathsf{d}_3) \cap var(\mathsf{d}_2) = \emptyset$ and therefore $var(\boldsymbol{G}') \cap var(\mathsf{d}_2) = \emptyset$. Then, by properties of $su(D)$ and since $\mathsf{d}_2 \in su(D)(\boldsymbol{G}'\gamma')$, there exists a derivation $\mathsf{d}_4 \in su(D)(\boldsymbol{G}')$ such that $var(\mathsf{d}_3) \cap var(\mathsf{d}_4) = var(\boldsymbol{G}')$ and

$$\partial_{\gamma'}(\mathsf{d}_4) = \mathsf{d}_2. \tag{A.18}$$

Then by definition of $\bowtie$ and by properties of $su(D)$, $\mathsf{d}_3 :: \mathsf{d}_4$ is a renamed apart (w.r.t. A) version of an element in $(D' \bowtie su(D))(A')$. Moreover, by definition of ∂ and of γ, $\partial_\gamma(\mathsf{d}_3 :: \mathsf{d}_4) \in (A \cdot (D' \bowtie su(D)))(A)$. Finally, the following hold:

$$\begin{aligned}
&\partial_\gamma(\mathsf{d}_3 :: \mathsf{d}_4)\\
&\quad = \partial_\gamma(\mathsf{d}_3) :: \partial_{\gamma'}(\mathsf{d}_4) \quad \text{(by Lemma 25(2) and by (A.17))}\\
&\quad = \mathsf{d} \quad \text{(by (A.15), (A.18) and (A.14)).}
\end{aligned}$$

($\sqsubseteq$): Let $\mathsf{d} \in (A \cdot (D' \bowtie su(D)))(A)$. If $\mathsf{d} \in (A \cdot D')(A)$ then, since by extensivity $A \cdot D' \sqsubseteq (A \cdot D') \bowtie su(D)$, $\mathsf{d} \in ((A \cdot D') \bowtie su(D))(A)$. Otherwise, by definition of $\bowtie$ and of $\cdot$, there exists a renamed apart (w.r.t. A) version d' of an element in $(D' \bowtie su(D))(A')$, for some atom $A' \leqslant A$, and there exists an idempotent substitution γ such that $A = first(\mathsf{d}')\gamma$ and

$$\partial_\gamma(\mathsf{d}') = \mathsf{d}. \tag{A.19}$$

Since d' is a renamed version of an element in $(D' \bowtie su(D))(A')$ and $\mathsf{d} \notin (A \cdot D')(A)$,

$$\mathsf{d}' = \mathsf{d}'_1 :: \mathsf{d}'_2, \tag{A.20}$$

where d'_1 is a renamed version of an element in $D'(A')$, $\boldsymbol{G}' = last(\mathsf{d}'_1)$ and, since $su(D)$ is closed under renaming, $\mathsf{d}'_2 \in su(D)(\boldsymbol{G}')$. Then, by definition of $\cdot$,

$$\partial_\gamma(\mathsf{d}'_1) \in (A \cdot D')(A). \tag{A.21}$$

Moreover, by (A.19)–(A.21) and since (by hypothesis) $\mathsf{d} \notin (A \cdot D')(A)$, $length(\partial_\gamma(\mathsf{d}')) > length(\partial_\gamma(\mathsf{d}'_1))$ and then $length(\partial_\gamma(\mathsf{d}'_1)) = length(\mathsf{d}'_1)$. Therefore, by (2) of Lemma 25,

$$\partial_\gamma(\mathsf{d}'_1 :: \mathsf{d}'_2) = \partial_\gamma(\mathsf{d}'_1) :: \partial_{\gamma'}(\mathsf{d}'_2), \tag{A.22}$$

where γ' is an idempotent substitution such that $last(\partial_\gamma(\mathsf{d}'_1)) = \boldsymbol{G}'\gamma' = (last(\mathsf{d}'_1))\gamma'$. Then, since $\mathsf{d}'_2 \in su(D)(\boldsymbol{G}')$, $\partial_{\gamma'}(\mathsf{d}'_2)$ is defined and, by properties of $su(D)$, $\partial_{\gamma'}(\mathsf{d}'_2) \in su(D)(\boldsymbol{G}'\gamma')$. Therefore, by definition of $\bowtie$, by (A.21) and (A.22), $\partial_\gamma(\mathsf{d}'_1) :: \partial_{\gamma'}(\mathsf{d}'_2) \in ((A \cdot D') \bowtie su(D))(A)$.

Now to prove the hypothesis it is sufficient to observe that, by (A.22), (A.20) and (A.19), $\partial_\gamma(\mathsf{d}'_1) :: \partial_{\gamma'}(\mathsf{d}'_2) = \partial_\gamma(\mathsf{d}'_1 :: \mathsf{d}'_2) = \partial_\gamma(\mathsf{d}') = \mathsf{d}$.

(2) By definition of $\sqsubseteq$, we have to prove that for any $\boldsymbol{G}' \in Goals$ $((D' \bowtie su(D)) \times \phi_{\boldsymbol{G}})(\boldsymbol{G}') \subseteq ((D' \times \phi_{\boldsymbol{G}}) \bowtie su(D))(\boldsymbol{G}')$. Let $\mathsf{d} \in ((D' \bowtie su(D)) \times \phi_{\boldsymbol{G}})(\boldsymbol{G}')$. Then by

definition of $\times$, $G' = (G_0, G)$ and there exists a renamed version d_1 of an element in $(D' \bowtie su(D))(G_0)$ such that $first(\mathsf{d}_1) = G_0$ and

$$\mathsf{d} = \mathsf{d}_1 \wedge G. \tag{A.23}$$

Now, by definition of $\bowtie$, two cases arise. If d_1 is a renamed version of an element in $D'(G_0)$ then, by definition of $\times$ and by (A.23), $\mathsf{d} \in (D' \times \phi_G)(G')$ and therefore, since $\bowtie$ is extensive, $\mathsf{d} \in ((D' \times \phi_G) \bowtie su(D))(G')$.

Otherwise, by definition of $\bowtie$ and since $su(D)$ is closed under renaming, $\mathsf{d}_1 = \mathsf{d}_3 :: \mathsf{d}_4$, where $\mathsf{d}_3 = G_0 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} B$ is a renamed version of an element in $D'(G_0)$, $B \neq \square$, $\mathsf{d}_4 \in su(D)(B)$ and $var(\mathsf{d}_3) \cap var(\mathsf{d}_4) = var(B)$. Let $\vartheta := \vartheta_1 \cdots \vartheta_k$. Then, by A.23 and by definition of $\wedge$, $\mathsf{d} = (\mathsf{d}_3 \wedge G) :: (\mathsf{d}_4 \wedge G\vartheta)$ and, by definition of $\times$, $\mathsf{d}_3 \wedge G \in (D' \times \phi_G)(G')$. Moreover, since $\mathsf{d}_4 \in su(D)(B)$, by definition of su and since $B \neq \square$, $\mathsf{d}_4 \wedge G\vartheta \in su(D)(B, G\vartheta)$. Finally, by definition of $\bowtie$, $\mathsf{d} \in ((D' \times \phi_G) \bowtie su(D))(G')$.

(3) We prove the two inclusions separately.

($\sqsubseteq$): Let $\mathsf{d} \in ((D' \times D'') \bowtie su(D))(G_0)$. We have two possibilities. If $\mathsf{d} \in (D' \times D'')(G_0)$ then, since $\bowtie$ is extensive and $\times$ is monotonic, $\mathsf{d} \in (D' \times (D'' \bowtie su(D)))(G_0)$. Otherwise, by definition of $\bowtie$ and since $su(D)$ is closed under renaming, there exist two derivations $\mathsf{d}' \in (D' \times D'')(G_0)$ and $\mathsf{d}'' \in su(D)(G)$ such that $last(\mathsf{d}') = G$,

$$\mathsf{d} = \mathsf{d}' :: \mathsf{d}'', \qquad var(\mathsf{d}') \cap var(\mathsf{d}'') = var(G). \tag{A.24}$$

Then, by definition of $\times$, there exist two goals G_0', G_0'' and two derivations $\mathsf{d}_1 = G_0' \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} G_k'$ and d_2, which are renamed versions of elements in $D'(G_0')$ and $D''(G_0'')$ respectively, such that $first(\mathsf{d}_2) = G_0''$, $G_0 = (G_0', G_0'')$ and $\mathsf{d}' = \mathsf{d}_1 \wedge \mathsf{d}_2$. Let $\vartheta := \vartheta_1 \cdots \vartheta_k$. Two cases arise

$G_k' \neq \square$: By definition of $\wedge$, $\mathsf{d}' = \mathsf{d}_1 \wedge G_0''$. Since (by hypothesis) d_2 is a renamed version of an element in $D''(G_0'')$, $D''(G_0'') \neq \emptyset$ and since (by definition of collection) $D''(G_0'')$ is a well-formed set of derivations, $G_0'' \in D''(G_0'')$. By (A.24), since $last(\mathsf{d}_1) = G_k' \neq \square$ and $last(\mathsf{d}') = (G_k', G_0''\vartheta)$,

$$\mathsf{d} = (\mathsf{d}_1 :: \mathsf{d}_3) \wedge G_0'', \tag{A.25}$$

where

$$\mathsf{d}_3 \in su(D)(G_k') \text{ is such that } \mathsf{d}'' = \mathsf{d}_3 \wedge G_0''\vartheta. \tag{A.26}$$

By definition of $\bowtie$, since $\mathsf{d}_1 :: \mathsf{d}_3$ is defined, since d_1 is a renamed version of an element in $D'(G_0')$ and by (A.26), $\mathsf{d}_1 :: \mathsf{d}_3$ is a renamed version of an element in $(D' \bowtie su(D))(G_0') = D'(G_0')$, where the last equality holds since (by hypothesis) $D' \bowtie su(D) = D'$. Therefore, by (A.25), by definition of $\times$ and since $G_0'' \in D''(G_0'')$, $\mathsf{d} = (\mathsf{d}_1 :: \mathsf{d}_3) \wedge G_0'' \in (D' \times D'')(G_0)$ and this contradicts the hypothesis.

$G_k' = \square$: By definition of $\wedge$,

$$\mathsf{d}' = \mathsf{d}_1 \wedge \mathsf{d}_2 = (\mathsf{d}_1 \wedge G_0'') :: \partial_\vartheta(\mathsf{d}_2). \tag{A.27}$$

We can assume that $length(\mathsf{d}_2) = length(\partial_\vartheta(\mathsf{d}_2))$, since $D''(G_0'')$ is a well-formed set of derivations. Then, by definition of ∂, by (A.24) and since by (A.27), $last(\partial_\vartheta(\mathsf{d}_2)) =$

$last(\mathsf{d}') = \boldsymbol{G}$, by (2) of Lemma 25 there exists an idempotent substitution δ such that $last(\partial_\vartheta(\mathsf{d}_2)) = \boldsymbol{G}''\delta = \boldsymbol{G}$, where $\boldsymbol{G}'' = last(\mathsf{d}_2)$.

Note that by (A.24) and since $var(\mathsf{d}') = var(\mathsf{d}_1) \cup var(\mathsf{d}_2)$, $var(\mathsf{d}_2) \cap var(clauses(\mathsf{d}'')) = \emptyset$ and $var(\boldsymbol{G}'') \cap var(clauses(\mathsf{d}'')) = \emptyset$. Then, by properties of $su(D)$, since $\mathsf{d}'' \in su(D)(\boldsymbol{G})$ and $\boldsymbol{G} = \boldsymbol{G}''\delta$, there exists $\mathsf{d}_3 \in su(D)(\boldsymbol{G}'')$ such that

$$\mathsf{d}'' = \partial_\delta(\mathsf{d}_3) \quad \text{and} \quad var(\mathsf{d}_3) \cap var(\mathsf{d}_2) = var(\boldsymbol{G}''). \tag{A.28}$$

Then $\mathsf{d}_2 :: \mathsf{d}_3$ is defined and, by (2) of Lemma 25 and since by construction $length(\mathsf{d}_2) = length(\partial_\vartheta(\mathsf{d}_2))$,

$$\partial_\vartheta(\mathsf{d}_2 :: \mathsf{d}_3) = \partial_\vartheta(\mathsf{d}_2) :: \partial_\delta(\mathsf{d}_3). \tag{A.29}$$

Then

$$\begin{aligned}
&\mathsf{d}\\
&= (\mathsf{d}_1 \wedge \mathsf{d}_2) :: \partial_\delta(\mathsf{d}_3) \quad \text{by (A.24), (A.27) and (A.28))}\\
&= ((\mathsf{d}_1 \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}_2)) :: \partial_\delta(\mathsf{d}_3) \quad \text{(by definition of } \wedge)\\
&= (\mathsf{d}_1 \wedge \boldsymbol{G}_0'') :: (\partial_\vartheta(\mathsf{d}_2) :: \partial_\delta(\mathsf{d}_3)) \quad \text{(since :: is associative)}\\
&= (\mathsf{d}_1 \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}_2 :: \mathsf{d}_3) \quad \text{(by (A.29))}\\
&= \mathsf{d}_1 \wedge (\mathsf{d}_2 :: \mathsf{d}_3) \quad \text{(by definition of } \wedge \text{ and of } \mathsf{d}_1).
\end{aligned}$$

By definition of $\bowtie$, since d_2 is a renamed version of an element in $D''(\boldsymbol{G}_0'')$, since $\mathsf{d}_3 \in su(D)(\boldsymbol{G}'')$ and $su(D)$ is closed under renaming, $\mathsf{d}_2 :: \mathsf{d}_3$ is a renamed version of an element in $(D'' \bowtie su(D))(\boldsymbol{G}_0'')$ such that $first(\mathsf{d}_2 :: \mathsf{d}_3) = \boldsymbol{G}_0''$. Finally, since d_1 is a renamed version of an element in $D'(\boldsymbol{G}_0')$ and $first(\mathsf{d}_1) = \boldsymbol{G}_0'$, by definition of $\times$, $\mathsf{d}_1 \wedge (\mathsf{d}_2 :: \mathsf{d}_3) \in (D' \times (D'' \bowtie su(D)))(\boldsymbol{G}_0', \boldsymbol{G}_0'')$.

($\sqsupseteq$): Let $\mathsf{d} \in (D' \times (D'' \bowtie su(D)))(\boldsymbol{G}_0)$. Then, by definition of $\times$, $\mathsf{d} = \mathsf{d}' \wedge \mathsf{d}''$, where $\boldsymbol{G}_0 = (\boldsymbol{G}_0', \boldsymbol{G}_0'')$ and d', d'' are renamed versions of elements in $D'(\boldsymbol{G}_0')$ and in $(D'' \bowtie su(D))(\boldsymbol{G}_0'')$, respectively, such that $first(\mathsf{d}') = \boldsymbol{G}_0'$ and $first(\mathsf{d}'') = \boldsymbol{G}_0''$. By definition of $\times$, two cases arise.

$last(\mathsf{d}') \neq \square$: In this case $\mathsf{d} = \mathsf{d}' \wedge \boldsymbol{G}_0'' \in (D' \times D'')(\boldsymbol{G}_0)$. Therefore, since $\bowtie$ is extensive, $\mathsf{d} \in ((D' \times D'') \bowtie su(D))(\boldsymbol{G}_0)$.

$last(\mathsf{d}') = \square$: By definition of $\bowtie$, we distinguish two cases. If d'' is a renamed version of an element in $D''(\boldsymbol{G}_0'')$, $\mathsf{d} = \mathsf{d}' \wedge \mathsf{d}'' \in (D' \times D'')(\boldsymbol{G}_0)$ and therefore, analogous to the previous case, $\mathsf{d} \in ((D' \times D'') \bowtie su(D))(\boldsymbol{G}_0)$. Otherwise assume that $\mathsf{d}' = \boldsymbol{G}_0' \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \square$ and let $\vartheta := \vartheta_1 \cdots \vartheta_k$. By definition of $\bowtie$ and since $su(D)$ is closed under renaming, $\mathsf{d}'' = (\mathsf{d}_1 :: \mathsf{d}_2)$, where $\mathsf{d}_1 = \boldsymbol{G}_0'' \xrightarrow[c_1']{\sigma_1} \cdots \xrightarrow[c_h']{\sigma_h} \boldsymbol{G}_h''$ is a renamed version of an element in $D''(\boldsymbol{G}_0)$ and $\mathsf{d}_2 \in su(D)(\boldsymbol{G}_h'')$. Moreover, by definition of $\wedge$,

$$\mathsf{d} = (\mathsf{d}' \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}'') = (\mathsf{d}' \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}_1 :: \mathsf{d}_2). \tag{A.30}$$

Now we have two possibilities

length$(\partial_\vartheta(\mathsf{d}_1)) <$ *length*(d_1): By (2) of Lemma 25, $\partial_\vartheta(\mathsf{d}_1 :: \mathsf{d}_2) = \partial_\vartheta(\mathsf{d}_1)$ and, therefore, by (A.30) and by definition of $\wedge$, $\mathsf{d} = (\mathsf{d}' \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}_1) = \mathsf{d}' \wedge \mathsf{d}_1 \in (D' \times D'')(\boldsymbol{G}_0) \subseteq ((D' \times D'') \bowtie su(D))(\boldsymbol{G}_0)$.

length$(\partial_\vartheta(\mathsf{d}_1)) =$ *length*(d_1): By (2) of Lemma 25 there exists an idempotent substitution δ such that $\boldsymbol{G}_h''\delta = last(\partial_\vartheta(\mathsf{d}_1))$ and

$$\partial_\vartheta(\mathsf{d}_1 :: \mathsf{d}_2) = \partial_\vartheta(\mathsf{d}_1) :: \partial_\delta(\mathsf{d}_2). \tag{A.31}$$

Moreover, since $\partial_\delta(\mathsf{d}_2)$ is defined, by properties of $su(D)$ and since $\mathsf{d}_2 \in su(D)(\boldsymbol{G}_h'')$,

$$\partial_\delta(\mathsf{d}_2) \in su(D)(\boldsymbol{G}_h''\delta). \tag{A.32}$$

Then the following facts hold:

$$\begin{aligned}
&\mathsf{d}\\
&= (\mathsf{d}' \wedge \boldsymbol{G}_0'') :: (\partial_\vartheta(\mathsf{d}_1) :: \partial_\delta(\mathsf{d}_2)) \quad \text{(by (A.30) and (A.31))}\\
&= ((\mathsf{d}' \wedge \boldsymbol{G}_0'') :: \partial_\vartheta(\mathsf{d}_1)) :: \partial_\delta(\mathsf{d}_2) \quad \text{(since :: is associative)}\\
&= (\mathsf{d}' \wedge \mathsf{d}_1) :: \partial_\delta(\mathsf{d}_2) \quad \text{(by definition of } \wedge).
\end{aligned}$$

Finally, by construction and (A.32), $(\mathsf{d}' \wedge \mathsf{d}_1) \in (D' \times D'')(\boldsymbol{G}_0)$, $\partial_\delta(\mathsf{d}_2) \in su(D)(\boldsymbol{G}_h''\delta)$. Therefore, since $(\mathsf{d}' \wedge \mathsf{d}_1) :: \partial_\delta(\mathsf{d}_2)$ is defined, by definition of $\bowtie$ and by the previous result, $\mathsf{d} = (\mathsf{d}' \wedge \mathsf{d}_1) :: \partial_\delta(\mathsf{d}_2) \in ((D' \times D'') \bowtie su(D))(\boldsymbol{G}_0)$. □

Lemma 26. *Let* $D \in \mathbb{P}\mathbb{C}$, $D_1, D_2 \in \mathbb{C}$ *and* $A \in Atoms$. *Then*
(1) $(A \cdot D_1) \bowtie pu(D) = A \cdot (D_1 \bowtie pu(D))$.
(2) $(D_1 \times D_2) \bowtie pu(D) \sqsubseteq (D_1 \bowtie pu(D)) \times (D_2 \bowtie pu(D))$.

Proof. The proof of (1) is analogous to that of (1) of Lemma 8 and hence omitted.

To prove (2), let $\mathsf{d} \in ((D_1 \times D_2) \bowtie pu(D))(\boldsymbol{G})$. If $\mathsf{d} \in (D_1 \times D_2)(\boldsymbol{G})$ then, since $\bowtie$ is extensive and $\times$ is monotonic, $\mathsf{d} \in ((D_1 \bowtie pu(D)) \times (D_2 \bowtie pu(D)))(\boldsymbol{G})$. Otherwise, by definition of $\bowtie$ and since $pu(D)$ is closed under renaming,

$$\mathsf{d} = \mathsf{d}' :: \mathsf{d}'', \tag{A.33}$$

where $\mathsf{d}' \in (D_1 \times D_2)(\boldsymbol{G})$, $last(\mathsf{d}') = \boldsymbol{B} \neq \square$, $\mathsf{d}'' \in pu(D)(\boldsymbol{B})$ and

$$var(\mathsf{d}') \cap var(\mathsf{d}'') = var(\boldsymbol{B}). \tag{A.34}$$

By definition of $\times$, $\mathsf{d}' = \mathsf{d}_1 \wedge \mathsf{d}_2$, where $\boldsymbol{G} = (\boldsymbol{G}_1, \boldsymbol{G}_2)$ and (for $i = 1,2$) d_i is a renamed version of an element in $D_i(\boldsymbol{G}_i)$ such that $first(\mathsf{d}_i) = \boldsymbol{G}_i$ and $var(\mathsf{d}_1) \cap var(\mathsf{d}_2) = var(\boldsymbol{G}_1) \cap var(\boldsymbol{G}_2)$. Let $\mathsf{d}_1 = \boldsymbol{G}_1 \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \boldsymbol{B}'$ and $\vartheta := \vartheta_1 \cdots \vartheta_k$. Two cases arise, we prove only the case $\boldsymbol{B}' \neq \square$ since the other ($\boldsymbol{B}' = \square$) is analogous. Then

$$\mathsf{d}' = \mathsf{d}_1 \wedge \boldsymbol{G}_2, \tag{A.35}$$

$\boldsymbol{B} = (\boldsymbol{B}', \boldsymbol{G}_2\vartheta)$ and $\mathsf{d}'' \in pu(D)(\boldsymbol{B}', \boldsymbol{G}_2\vartheta)$. Moreover, by 21 and since (by Lemma 6) $\times$ is associative,

$$\mathsf{d}'' = \mathsf{d}_3 \wedge \mathsf{d}_4, \tag{A.36}$$

where $\mathsf{d}_3 \in pu(D)(\boldsymbol{B}')$ and $\mathsf{d}_4 \in pu(D)(\boldsymbol{G}_2\vartheta)$. By (A.34), $var(\mathsf{d}_1) \cap var(\mathsf{d}_3) = var(\boldsymbol{B}')$ and therefore, by definition of $\bowtie$ and since pu is closed under renaming,

$$\mathsf{d}_1 :: \mathsf{d}_3 \text{ is a renamed version of an element in } (D_1 \bowtie pu(D))(\boldsymbol{G}_1). \tag{A.37}$$

Moreover, since $\mathsf{d}_4 \in pu(D)(\boldsymbol{G}_2\vartheta)$ and since, by our hypothesis on variables, $var(\mathsf{d}_4) \cap var(\boldsymbol{G}_2) \subseteq var(\boldsymbol{G}_2\vartheta)$, by properties of $pu(D)$ there exists

$$\mathsf{d}_5 \in pu(D)(\boldsymbol{G}_2) \text{ such that } \mathsf{d}_4 = \partial_\vartheta(\mathsf{d}_5). \tag{A.38}$$

Now observe that, since $D_2(\boldsymbol{G}_2) \neq \emptyset$ is a well-formed set of derivations, $\boldsymbol{G}_2 \in D_2(\boldsymbol{G}_2)$ and therefore, by definition of $\bowtie$,

$$\mathsf{d}_5 = \boldsymbol{G}_2 :: \mathsf{d}_5 \in (D_2 \bowtie pu(D))(\boldsymbol{G}_2). \tag{A.39}$$

By our hypothesis on variables, $var(\mathsf{d}_1 :: \mathsf{d}_3) \cap var(\mathsf{d}_5) = var(\boldsymbol{G}_1) \cap var(\boldsymbol{G}_2)$. Therefore, by definition of $\times$, by (A.37) and (A.39) and since $\boldsymbol{G} = (\boldsymbol{G}_1, \boldsymbol{G}_2)$, $(\mathsf{d}_1 :: \mathsf{d}_3) \wedge \mathsf{d}_5 \in ((D_1 \bowtie pu(D)) \times (D_2 \bowtie pu(D)))(\boldsymbol{G})$. Finally, since (by hypothesis) $last(\mathsf{d}_1) \neq \square$,

$$\begin{aligned}
&(\mathsf{d}_1 :: \mathsf{d}_3) \wedge \mathsf{d}_5 \\
&\quad = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: (\mathsf{d}_3 \wedge \partial_\vartheta(\mathsf{d}_5)) \quad \text{(by definition of } \wedge) \\
&\quad = (\mathsf{d}_1 \wedge \boldsymbol{G}_2) :: (\mathsf{d}_3 \wedge \mathsf{d}_4) \quad \text{(by (A.38))} \\
&\quad = \mathsf{d}' :: \mathsf{d}'' \quad \text{(by (A.35) and (A.36))} \\
&\quad = \mathsf{d} \quad \text{(by (A.33)).} \qquad \square
\end{aligned}$$

Proof of Lemma 16. (1) We have to prove that $\forall \boldsymbol{G} \in Goals.(pu(D) \bowtie pu(D'))(\boldsymbol{G}) \subseteq (pu(D \bowtie pu(D')))(\boldsymbol{G})$. The proof is by structural induction on $\boldsymbol{G}$. If $\boldsymbol{G} = \square$ then, by (21), $(pu(D) \bowtie pu(D'))(\square) = \{\square\} = (pu(D \bowtie pu(D')))(\square)$.

Otherwise let $\boldsymbol{G} := (A, \boldsymbol{G}')$ and $\mathsf{d} \in (pu(D) \bowtie pu(D'))(\boldsymbol{G})$. Two cases arise.

$\mathsf{d} \in pu(D)(\boldsymbol{G})$: In this case, since $\bowtie$ is extensive and $\cdot$ and $\times$ are monotonic, $pu(D) \sqsubseteq pu(D \bowtie pu(D'))$ and then $\mathsf{d} \in pu(D \bowtie pu(D'))(\boldsymbol{G})$.

$\mathsf{d} \notin pu(D)(\boldsymbol{G})$: Since $pu(D')$ is closed under renaming,

$$\mathsf{d} = \mathsf{d}_1 :: \mathsf{d}_2, \tag{A.40}$$

where $\mathsf{d}_1 \in pu(D)(\boldsymbol{G})$, $last(\mathsf{d}_1) = \boldsymbol{B} \neq \square$ and $\mathsf{d}_2 \in pu(D')(\boldsymbol{B})$. By (21) and since pu is closed under renaming, $\mathsf{d}_1 = \mathsf{d}_3 \wedge \mathsf{d}_4$, where $\mathsf{d}_3 \in (A \cdot D)(A)$ and $\mathsf{d}_4 \in pu(D)(\boldsymbol{G}')$. Then, by definition of $\times$ and by (2) of Lemma 26, $\mathsf{d}_1 :: \mathsf{d}_2 \in (((A \cdot D) \times pu(D)) \bowtie pu(D'))(\boldsymbol{G}) \subseteq (((A \cdot D) \bowtie pu(D')) \times (pu(D) \bowtie pu(D')))(\boldsymbol{G})$. Therefore, by A.40, by definition of $\times$ and since $\boldsymbol{G} = (A, \boldsymbol{G}')$, there exist two renamed versions d_5 and d_6 of elements in $((A \cdot D) \bowtie pu(D'))(A)$ and in $(pu(D) \bowtie pu(D'))(\boldsymbol{G}')$, respectively, such that $\mathsf{d} = \mathsf{d}_5 \wedge \mathsf{d}_6$. By inductive hypothesis, d_6 is a renamed version of an element in $pu(D \bowtie pu(D'))(\boldsymbol{G}')$. Moreover, by (1) of Lemma 26 and since d_5 is a renamed version of an element in $((A \cdot D) \bowtie pu(D'))(A)$, d_5 is a renamed version of an element in $((A \cdot D) \bowtie pu(D'))(A)$. Finally, by definition of $\times$ and pu and since $\mathsf{d} = \mathsf{d}_5 \wedge \mathsf{d}_6$, $\mathsf{d} \in ((A \cdot (D \bowtie pu(D'))) \times pu(D \bowtie pu(D')))(\boldsymbol{G}) \subseteq pu(D \bowtie pu(D'))(\boldsymbol{G})$.

(2) The proof is by induction on h. The case $h=0$ is straightforward, since $pu^0(D) = \phi = pu_0(D)$. Otherwise the proof is by structural induction on $\boldsymbol{G}$.

$\boldsymbol{G} = \square$: By (23) and (22), $pu^h(D)(\square) = \{\square\} = pu_1(D)(\square)$.

$\boldsymbol{G} = A, \boldsymbol{G}'$: Let $\mathsf{d} \in pu^h(D)(\boldsymbol{G}) = (pu(D \bowtie pu^{h-1}(D)))(\boldsymbol{G})$. Then by (23),

$$\mathsf{d} = \mathsf{d}_1 \wedge \mathsf{d}_2, \text{ where } \mathsf{d}_1 \in (A \cdot (D \bowtie pu^{h-1}(D)))(A) \tag{A.41}$$

and $\mathsf{d}_2 \in pu^h(D)(\boldsymbol{G}')$. By inductive hypothesis, $\mathsf{d}_2 \in \sum\{pu_k(D)\}_{k\geqslant 0}(\boldsymbol{G}')$ and therefore there exists $m \geqslant 0$ such that

$$\mathsf{d}_2 \in pu_m(D)(\boldsymbol{G}'). \tag{A.42}$$

Moreover,

$$\begin{aligned}
&A \cdot (D \bowtie pu^{h-1}(D)) \\
&\quad = (A \cdot D) \bowtie pu^{h-1}(D) \quad \text{(by Lemma 26(1))} \\
&\quad \sqsubseteq (A \cdot D) \bowtie \sum\{pu_k(D)\}_{k\geqslant 0} \quad \text{(by inductive hypothesis)} \\
&\quad \sqsubseteq pu(D) \bowtie \sum\{pu_k(D)\}_{k\geqslant 0} \quad \text{(by (21))} \\
&\quad = \sum\{pu(D) \bowtie pu_k(D)\}_{k\geqslant 0} \quad \text{(by Lemma 7)} \\
&\quad = \sum\{pu_k(D)\}_{k\geqslant 0} \quad \text{(by (22))}.
\end{aligned}$$

Then, by (A.41), there exists $l \geqslant 0$ such that

$$\mathsf{d}_1 \in pu_l(D)(A). \tag{A.43}$$

Now, by definition of $\wedge$, we have two possibilities.

$last(\mathsf{d}_1) \neq \square$: In this case $\mathsf{d} = \mathsf{d}_1 \wedge \boldsymbol{G}'$. Since $\mathsf{d}_2 \in pu_m(D)(\boldsymbol{G}')$, for any predicate symbol p occurring in $\boldsymbol{G}'$, $D(p(\boldsymbol{x})) \neq \emptyset$. Then $\boldsymbol{G}' \subset pu_l(D)(\boldsymbol{G}')$ and therefore, by (21), (A.43) and by definition of $\bowtie$, $\mathsf{d} = \mathsf{d}_1 \wedge \boldsymbol{G}' \in pu_l(D)(\boldsymbol{G}) \subseteq \sum\{pu_k(D)\}_{k\geqslant 0}(\boldsymbol{G})$.

$last(\mathsf{d}_1) = \square$: Let $\mathsf{d}_1 := A \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \square$ and $\vartheta := \vartheta_1 \cdots \vartheta_k$. By definition of $\wedge$ and by (A.41),

$$\mathsf{d} = (\mathsf{d}_1 \wedge \boldsymbol{G}') :: \partial_\vartheta(\mathsf{d}_2). \tag{A.44}$$

Analogous to the previous case, by using (A.43),

$$\mathsf{d}_1 \wedge \boldsymbol{G}' \in pu_l(D)(\boldsymbol{G}). \tag{A.45}$$

Moreover, by (A.44), $\partial_\vartheta(\mathsf{d}_2)$ is defined and, by (A.42), $\mathsf{d}_2 \in pu_m(D)(\boldsymbol{G}')$. Then, by properties of pu and by a straightforward inductive argument, $\partial_\vartheta(\mathsf{d}_2) \in pu_m(D)(\boldsymbol{G}'\vartheta)$. Then, by definition of $\bowtie$, by (A.44) and (A.45), $\mathsf{d} = (\mathsf{d}_1 \wedge \boldsymbol{G}') :: \partial_\vartheta(\mathsf{d}_2) \in (pu_l(D) \bowtie pu_m(D))(\boldsymbol{G})$ and therefore, by Lemma 6 and by a straightforward inductive argument, $\mathsf{d} \in pu_{l+m}(D)(\boldsymbol{G})$. $\square$

Proof of Lemma 18. (1) By definition of $\sqsubseteq$ it is sufficient to prove that, for any $\boldsymbol{G} \in Goals$, $su(D)(\boldsymbol{G}) \subseteq pu(D + Id_{\mathbb{I}})(\boldsymbol{G})$. We distinguish two cases. If $\boldsymbol{G} = \square$, $su(D)(\square)$ is

not defined and then the hypothesis follows trivially. Otherwise let $\boldsymbol{G} = (A, \boldsymbol{G}')$ and observe that, by 21, $Id_{\mathbb{C}} \sqsubseteq pu(D + Id_{\mathbb{I}})$. Then the following facts hold:

$$
\begin{aligned}
&su(D)(\boldsymbol{G})\\
&\quad = ((A \cdot D) \times Id_{\mathbb{C}})(\boldsymbol{G}) \quad \text{(by (8))}\\
&\quad \subseteq ((A \cdot D) \times pu(D + Id_{\mathbb{I}}))(\boldsymbol{G}) \quad \text{(by previous observation)}\\
&\quad \subseteq ((A \cdot (D + Id_{\mathbb{I}})) \times pu(D + Id_{\mathbb{I}}))(\boldsymbol{G}) \quad \text{(since } \times \text{ is monotonic)}\\
&\quad = pu(D + Id_{\mathbb{I}})(\boldsymbol{G}) \quad \text{(by definition of } pu).
\end{aligned}
$$

(2) We prove (by induction on n) that, for any $\boldsymbol{G} = A_1, \ldots, A_n \in Goals$ and $n \geqslant 0$, $\mathscr{G}[\![\boldsymbol{G}]\!]_D \sqsubseteq Id_{\mathbb{C}} + su_n(D)$. Then the hypothesis follows by (21).

If $n = 0$ then the hypothesis follows trivially, since $\boldsymbol{G} = \square$ and, by definition of $\mathscr{G}$, $\mathscr{G}[\![\boldsymbol{G}]\!]_D = \phi_{\square} \sqsubseteq Id_{\mathbb{C}}$. Otherwise let $\boldsymbol{G} := A, \boldsymbol{G}'$. Then

$$
\begin{aligned}
&\mathscr{G}[\![A, \boldsymbol{G}']\!]_D\\
&\quad = (A \cdot D) \times \mathscr{G}[\![\boldsymbol{G}']\!]_D \quad \text{(by definition of } \mathscr{G} \text{ and of } \mathscr{A})\\
&\quad \sqsubseteq (A \cdot D) \times (Id_{\mathbb{C}} + su_{n-1}(D)) \quad \text{(by inductive hypothesis).}
\end{aligned}
$$

Let $\mathsf{d} \in \mathscr{G}[\![\boldsymbol{G}]\!]_D(\boldsymbol{G})$. By previous result and by definition of $\times$, $\mathsf{d} = \mathsf{d}_1 \wedge \mathsf{d}_2$, where $\mathsf{d}_1 \in (A \cdot D)(A)$ and $\mathsf{d}_2 \in (Id_{\mathbb{C}} + su_{n-1}(D))(\boldsymbol{G}')$. Two cases arise.

$last(\mathsf{d}_1) \neq \square$: In this case, by definition of $\wedge$, $\mathsf{d} = \mathsf{d}_1 \wedge \boldsymbol{G}' \in ((A \cdot D) \times Id_{\mathbb{C}})(\boldsymbol{G})$ and therefore, by (8), $\mathsf{d} \in su(D)(\boldsymbol{G})$. Then, by definition of $\bowtie$ and of $+$, $\mathsf{d} \in (Id_{\mathbb{C}} + su_n(D))(\boldsymbol{G})$.

$last(\mathsf{d}_1) = \square$: Let $\mathsf{d}_1 := A \xrightarrow[c_1]{\vartheta_1} \cdots \xrightarrow[c_k]{\vartheta_k} \square$ and $\vartheta := \vartheta_1 \cdots \vartheta_k$. By definition of $\wedge$, $\mathsf{d} = (\mathsf{d}_1 \wedge \boldsymbol{G}') :: \partial_{\vartheta}(\mathsf{d}_2)$. Then, since $\mathsf{d}_1 \in (A \cdot D)(A)$, $\mathsf{d}_1 \wedge \boldsymbol{G}' \in ((A \cdot D) \times Id_{\mathbb{C}})(\boldsymbol{G})$ and, therefore, by (8), $\mathsf{d}_1 \wedge \boldsymbol{G}' \in su(D)(\boldsymbol{G})$. Moreover, since $\partial_{\vartheta}(\mathsf{d}_2)$ is defined and $\mathsf{d}_2 \in (Id_{\mathbb{C}} + su_{n-1}(D))(\boldsymbol{G}')$, by properties of su and by a straightforward inductive argument, $\partial_{\vartheta}(\mathsf{d}_2) \in (Id_{\mathbb{C}} + su_{n-1}(D))(\boldsymbol{G}'\vartheta)$. By previous results and by definition of $\bowtie$,

$$
\mathsf{d} = (\mathsf{d}_1 \wedge \boldsymbol{G}') :: \partial_{\vartheta}(\mathsf{d}_2) \in (su(D) \bowtie (Id_{\mathbb{C}} + su_{n-1}(D)))(\boldsymbol{G}). \tag{A.46}
$$

Finally, observe that, since $n \geqslant 1$, $su(D) \sqsubseteq su_n(D)$. Then

$$
\begin{aligned}
&su(D) \bowtie (Id_{\mathbb{C}} + su_{n-1}(D))\\
&\quad = (su(D) \bowtie Id_{\mathbb{C}}) + (su(D) \bowtie su_{n-1}(D)) \quad \text{(by Lemma 7)}\\
&\quad = su(D) + (su(D) \bowtie su_{n-1}(D)) \quad \text{(since } D \bowtie Id_{\mathbb{C}} = D)\\
&\quad = su_n(D) \quad \text{(by (13) and previous observation).}
\end{aligned}
$$

Therefore, by (A.46), $\mathsf{d} \in su_n(D)(\boldsymbol{G}) \subseteq (Id_{\mathbb{C}} + su_n(D))(\boldsymbol{G})$. $\square$

References

[1] K.R. Apt, Introduction to logic programming, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, vol. B: Formal Models and Semantics, Elsevier and The MIT Press, Amsterdam and Cambridge, 1990, pp. 495–574.

[2] A. Bossi, M. Gabbrielli, G. Levi, M. Martelli, The s-semantics approach: Theory and applications, J. Logic Programm. 19–20 (1994) 149–197.

[3] M. Comini, G. Levi, M.C. Meo, Compositionality of *SLD*-derivations and their abstractions, in: J. Lloyd (Ed.), Proc. 1995 Internat. Symp. on Logic Programming, The MIT Press, Cambridge, 1995, pp. 561–575.

[4] M. Comini, G. Levi, M.C. Meo, A theory of observables for logic programs, 1996, submitted.

[5] M. Comini, G. Levi, G. Vitiello, Efficient detection of incompleteness errors in the abstract debugging of logic programs, in: M. Ducassé (Ed.), Proc. 2nd Internat. Workshop on Automated and Algorithmic Debugging, AADEBUG '95, 1995.

[6] P. Cousot, R. Cousot, Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints, In: Proc. 4th ACM Symp. Principles of Programming Languages, 1977, pp. 238–252.

[7] S.K. Debray, P. Mishra, Denotational and operational semantics for prolog, J. Logic Programm. 5 (1988) 61–91.

[8] E. Eder, Properties of substitutions and unification, J. Symbolic Comput. 1 (1985) 31–46.

[9] M. Falaschi, G. Levi, M. Martelli, C. Palamidessi, Declarative modeling of the operational behavior of logic languages, Theoret. Comput. Sci. 69(3) (1989) 289–318.

[10] M. Gabbrielli, G. Levi, M.C. Meo, Observational equivalences for logic programs, in: K. Apt (Ed.), Proc. Joint Internat. Conf. and Symp. on Logic Programming, The MIT Press, Cambridge, 1992, pp. 131–145.

[11] M. Gabbrielli, G. Levi, M.C. Meo, Resultants semantics for PROLOG, J. Logic Comput. 6(4) (1996) 491–521.

[12] N.D. Jones, A. Mycroft, Stepwise development of operational and denotational semantics for PROLOG, in: Sten-Åke Tärnlund (Ed.), Proc. Second Internat. Conf. on Logic Programming, 1984, pp. 281–288.

[13] N.D. Jones, H. Søndergaard, A semantics-based framework for the abstract interpretation of PROLOG, in: S. Abramsky, C. Hankin (Eds.), Abstract Interpretation of Declarative Languages, Ellis Horwood, Chichester, 1987, pp. 123–142.

[14] R. Kemp, G. Ringwood, An algebraic framework for the abstract interpretation of logic programs, in: S.K. Debray, M. Hermenegildo (Eds.), Proc. North American Conf. on Logic Programming '90, The MIT Press, Cambridge, 1990, pp. 506–520.

[15] R. Kemp, G. Ringwood, Reynolds base, Clark Models and Heyting semantics of logic programs, submitted.

[16] J.L. Lassez, M.J. Maher, K. Marriott, Unification revisited, in: J. Minker (Ed.), Foundations of Deductive Databases and Logic Programming, Morgan Kaufmann, Los Altos, CA, 1988, pp. 587–625.

[17] J.W. Lloyd, Foundations of Logic Programming, 2nd ed. Springer, Berlin, 1987.